Enabling Reinforcement Learning in Stochastic Environments Through Noisy Dueling Networks and Modular Hierarchical Architectures

Chee Jun Kit
Department of Computer Science
University of Nottingham Malaysia

Introduction

Reinforcement learning (RL) has shown significant potential in solving complex control problems, with algorithms like Deep Q-Learning (DQN) achieving remarkable success in deterministic environments (Sadavarte et al., 2021). However, real-world scenarios often exhibit stochasticity, where random environmental factors such as wind disturbances, sensor inaccuracies, or mechanical failures disrupt the agent's learning process. Addressing these challenges is critical to enhancing the robustness and generalization of RL algorithms in real-world applications.

One prominent limitation in existing RL research is the focus on deterministic episodic environments (Han et al., 2022; Sadavarte et al., 2021). While these setups simplify algorithm evaluation, they fail to capture the unpredictability inherent in many real-world tasks. Furthermore, current RL approaches often rely on monolithic architectures, neglecting the benefits of modular learning frameworks (Steccanella et al., 2020). This lack of modularity can hinder an agent's ability to adapt efficiently to distinct challenges, such as handling core tasks and managing environmental noise separately.

To address these gaps, this research explores two central questions. First, how can reinforcement learning algorithms like DQN be adapted to handle stochastic environments effectively, and does this adaptation lead to better generalization compared to deterministic setups? Second, can breaking the learning process into specialized modules—such as one for core task management and the other for noise handling—improve performance and learning efficiency?

To tackle these questions, this work introduces two novel approaches. The first is NoisyDueling-DQN, which incorporates Noisy Dueling Networks to enhance exploration and manage environmental noise (Han et al., 2022). The second is a Hierarchical Reinforcement Learning (HRL) framework, referred to as HRL-NoisyDueling-DQN, which decomposes the learning process into specialized modules overseen by a high-level manager policy (Steccanella et al., 2020). These approaches are evaluated in both deterministic and stochastic settings using CartPole-v1 as a control environment and LunarLander-v3 as the primary environment.

The results demonstrate that NoisyDueling-DQN significantly improves robustness to noise compared to single-agent DQN, particularly in stochastic environments where single-agent DQN fails to converge (Han et al., 2022; Sadavarte et al., 2021). Moreover, the HRL-NoisyDueling-DQN framework outperforms single-agent DQN in both learning speed and adaptability, effectively solving stochastic LunarLander-v3 in fewer episodes (Steccanella et al., 2020). These findings highlight the importance of noise reduction techniques and modular learning for enhancing RL performance in complex, unpredictable environments.

This study contributes a systematic approach to adapting RL algorithms for stochastic environments and introduces modular frameworks to improve learning efficiency. While inspired by NROWAN-DQN

principles (Han et al., 2022), this implementation emphasizes noise management through Noisy Dueling Networks and modular learning using HRL. Future work could explore incorporating weighted attention mechanisms to further enhance noise resistance.

Background

Reinforcement learning (RL) is a subfield of machine learning that focuses on training agents to make sequential decisions by interacting with an environment (Sadavarte et al., 2021). Among its algorithms, Deep Q-Learning (DQN) has been widely adopted for solving control problems due to its ability to approximate value functions using deep neural networks. Applications of DQN span across domains like robotics, gaming, and autonomous navigation. Despite its successes, traditional DQN struggles in environments with stochasticity, where unpredictability challenges the robustness of learned policies (Han et al., 2022).

In exploring this problem, the literature reveals several critical gaps. Most RL studies evaluate algorithms in deterministic environments, such as CartPole-v1 and LunarLander-v3, where environmental parameters remain fixed (Sadavarte et al., 2021). This focus, while useful for initial validation, limits the applicability of findings to real-world scenarios where randomness, such as wind fluctuations or sensor noise, is prevalent. For instance, key studies on DQN have demonstrated its effectiveness in deterministic setups but do not explore its resilience to stochastic perturbations, leaving its adaptability in real-world applications untested (Han et al., 2022).

Another limitation in the literature is the lack of modular approaches in RL. Monolithic architectures dominate RL research, where a single agent learns all aspects of the problem. This design is suboptimal for environments with distinct challenges, such as balancing core control tasks and mitigating noise. Modular frameworks, which allocate specialized roles to different components of an agent, are underexplored in the context of RL (Steccanella et al., 2020). Such architectures could potentially improve learning efficiency and adaptability by allowing focused learning on specific subproblems.

To address these gaps, this research builds on existing work while introducing novel contributions. The focus on DQN stems from its foundational role in RL and its limitations in handling stochastic environments. Furthermore, the exploration of modular learning frameworks, inspired by hierarchical approaches in robotics and decision-making systems, represents an innovative direction that addresses the shortcomings of monolithic designs (Steccanella et al., 2020).

The motivation to explore stochasticity and modularity arises not only from theoretical insights but also from practical considerations. In real-world applications, such as robotic control or autonomous vehicles, environments are rarely deterministic. Wind disturbances, terrain irregularities, or sensor inaccuracies introduce unpredictability that conventional RL algorithms are ill-equipped to handle (Sadavarte et al., 2021). By bridging these gaps, this research aims to advance the robustness and generalizability of RL systems, paving the way for their broader adoption in real-world settings.

Methods

This research implements and evaluates two novel reinforcement learning approaches—NoisyDueling-DQN and HRL-NoisyDueling-DQN—designed to address the challenges of stochastic environments and modular learning. The implementation was conducted in Python using the OpenAI's Gymnasium library for simulation environments, with CartPole-v1 and LunarLander-v3 selected as the control and primary environments, respectively.

NoisyDueling-DQN

The first approach, NoisyDueling-DQN, extends traditional DQN by incorporating Noisy Dueling Networks. This architecture consists of two separate streams—value and advantage—enabling the agent to differentiate between actions that are critical to performance and those that are not. The NoisyLinear layers in the network introduce parameterized noise during exploration, dynamically adjusted throughout training to improve stability and convergence in stochastic settings. Noise injection into the environment was achieved using a custom NoiseInjector wrapper, which modifies parameters like gravity, wind power, and turbulence in LunarLander-v3 to simulate real-world unpredictability. A dynamic sigma adjustment mechanism ensures that the exploration noise decays as the agent learns, further enhancing robustness.

HRL-NoisyDueling-DQN

The second approach introduces an HRL framework. This design decomposes the learning process into specialized modules managed by a high-level policy. The HRL agent consists of three components: Module 1, which focuses on core tasks like balance or engine control; Module 2, which specializes in handling stochasticity; and a manager, which dynamically selects the appropriate module based on the current state. Each module is implemented as a NoisyDueling-DQN agent, trained independently with tailored hyperparameters. The manager itself is also a DQN agent with an action space corresponding to the available modules. This hierarchical design allows each module to focus on specific aspects of the problem, improving learning efficiency and adaptability in complex environments.

Experimental Design

Training was conducted in deterministic and stochastic modes for both CartPole-v1 and LunarLander-v3. Deterministic setups used default environmental parameters, while stochastic setups introduced noise via the NoiseInjector. A system ensured repeatability across all experiments. The performance of the proposed algorithms was compared against a baseline single-agent DQN to evaluate robustness and efficiency. Metrics such as the number of episodes required to solve the environment, total rewards, and running averages were tracked. Parameter tuning for HRL-NoisyDueling-DQN was performed iteratively, as initial configurations failed to solve the environments due to the complexity introduced by modular learning.

Results

Baseline: Single-Agent DQN
While this study focuses on advanced algorithms such as **NoisyDueling DQN** and **HRL-NoisyDueling-DQN**, the foundational approach—**Single-Agent DQN**—was implemented and tested during earlier stages of development. Although the experimental results for Single-Agent DQN are not included in this study due to the lack of saved files, it served as the basis for the current implementations.

From my prior work, **Single-Agent DQN** demonstrated expected performance in deterministic environments, solving **CartPole-v1** in approximately 350–400 episodes and **LunarLander-v3** in around 850–900 episodes, However, it proved ineffective in stochastic settings, unable to handle environmental unpredictability.

HRL-NoisyDueling-DQN and NoisyDueling-DQN
The experimental results confirm that NoisyDueling-DQN outperformed the DQN baseline in both deterministic and stochastic settings, demonstrating improved robustness through noise reduction techniques. For CartPole-v1, NoisyDueling-DQN solved the deterministic environment in 241 episodes and the stochastic environment in 193 episodes. For LunarLander-v3, it achieved convergence in 484 episodes in

deterministic setups and 561 episodes in stochastic ones. Programs without HRL, such as NoisyDueling-DQN, were approximately three times faster than their HRL counterparts in both CartPole and LunarLander environments.

The HRL-NoisyDueling-DQN framework, while slower, achieved the best overall results in terms of adaptability and performance. Leveraging modularity to handle noise effectively, it solved the deterministic setup of CartPole-v1 in 219 episodes and the stochastic setup in 227 episodes. For LunarLander-v3, HRL-NoisyDueling-DQN converged in 480 episodes in deterministic setups and 513 episodes in stochastic ones. These results highlight the trade-off between computational cost and robustness, with HRL approaches excelling in managing noise through specialized modules.

Visualization of Results
Graphs detailing the training progress for NoisyDueling-DQN and HRL-NoisyDueling-DQN, including reward trends and convergence rates, are provided in the appendices. These visuals illustrate the significant performance improvements in stochastic environments and the stability achieved through modular learning.

Discussion

The results validate the effectiveness of noise reduction techniques and modularity in improving RL performance in stochastic environments. NoisyDueling-DQN enhances exploration and robustness through NoisyLinear layers, but its performance in high-noise scenarios remains limited. HRL-NoisyDueling-DQN addresses these challenges by separating core task management and noise handling into specialized modules, significantly improving adaptability and learning efficiency.

While the hierarchical framework increases complexity and computational costs, it consistently outperforms single-agent DQN in both deterministic and stochastic environments. These findings highlight the importance of modular designs for RL systems in real-world applications.

Conclusion

This study demonstrates that integrating noise reduction techniques and modular learning frameworks can significantly enhance reinforcement learning (RL) performance in stochastic environments. The NoisyDueling-DQN improves robustness by leveraging NoisyLinear layers and a dueling network architecture, while the HRL-NoisyDueling-DQN framework further enhances adaptability by decomposing the learning process into specialized modules. The hierarchical approach allows the agent to focus on distinct challenges, such as core task management and noise handling, resulting in faster convergence and improved performance compared to single-agent DQN.

By addressing key gaps in the literature, this research highlights the potential of modular designs and noise management techniques in advancing RL systems. The findings lay a foundation for developing RL agents capable of operating effectively in complex, unpredictable real-world scenarios.

Future Work

Future research could explore integrating attention mechanisms to enhance noise resistance and extending the framework to more complex environments with higher-dimensional state and action spaces. Additionally, optimizing the computational efficiency of HRL-NoisyDueling-DQN would make it more practical for large-scale applications. A promising direction would involve evolving the framework into a full-fledged HRL-NROWAN-DQN by incorporating noise-resistant weighted attention mechanisms. This would enable the agent to dynamically prioritize critical features, further improving robustness in stochastic environments and advancing its adaptability for real-world challenges.

References:

Han, S., Zhou, W., Lu, J., Liu, J., & Lü, S. (2022). NROWAN-DQN: A stable noisy network with noise reduction and online weight adjustment for exploration. *Expert Systems with Applications, 203,* 117343. https://doi.org/10.1016/j.eswa.2022.117343

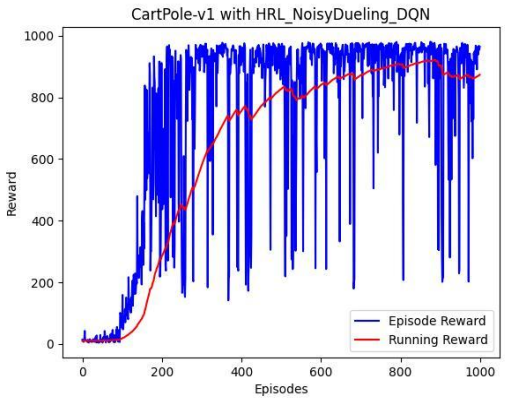
Sadavarte, R. S., Raj, R., & Babu, B. S. (2021). Solving the Lunar Lander problem using reinforcement learning. In *2021 IEEE International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS)* (pp. 1–7). IEEE. https://doi.org/10.1109/CSITSS54238.2021.9682970

Steccanella, L., Totaro, S., Allonsius, D., & Jonsson, A. (2020). Hierarchical reinforcement learning for efficient exploration and transfer. *arXiv preprint arXiv:2011.06335*. https://arxiv.org/abs/2011.06335

Appendices:

```
Enter random seed (e.g., 42): 10
Choose mode: Deterministic (D) or Stochastic (S): d
Do you want to use the HRL agent? (Y/N): y
```

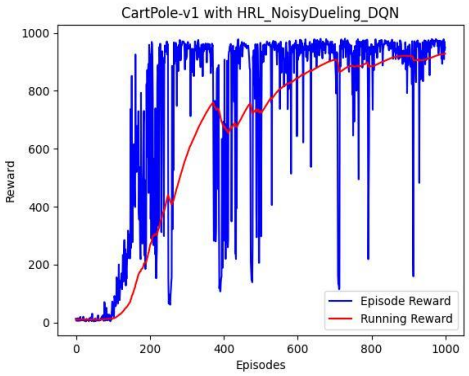
Input used for HRL_NoisyDueling_DQN for Deterministic CartPole-v1



HRL_NoisyDueling_DQN for Deterministic CartPole-v1

```
Enter random seed (e.g., 42): 10
Choose mode: Deterministic (D) or Stochastic (S): s
Enter gravity noise (e.g., 0.1 for 10% increase): 0.1
Do you want to use the HRL agent? (Y/N): y
```

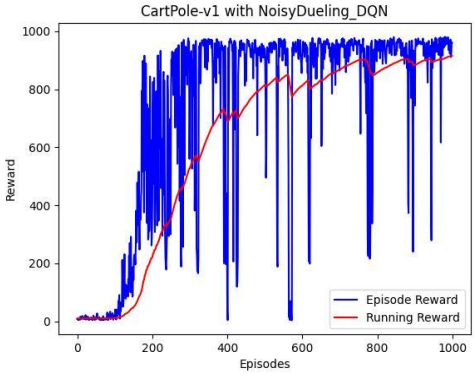
Input used for HRL_NoisyDueling_DQN for Stochastic CartPole-v1



HRL_NoisyDueling_DQN for Stochastic CartPole-v1

```
Enter random seed (e.g., 42): 10
Choose mode: Deterministic (D) or Stochastic (S): d
Do you want to use the HRL agent? (Y/N): n
```

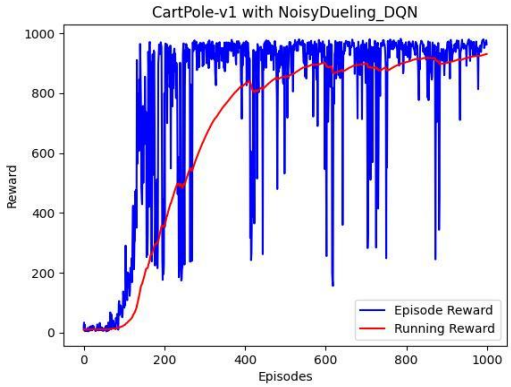
Input used for NoisyDueling_DQN for Deterministic CartPole-v1



NoisyDueling_DQN for Deterministic CartPole-v1

```
Enter random seed (e.g., 42): 10
Choose mode: Deterministic (D) or Stochastic (S): s
Enter gravity noise (e.g., 0.1 for 10% increase): 0.1
Do you want to use the HRL agent? (Y/N): n
```

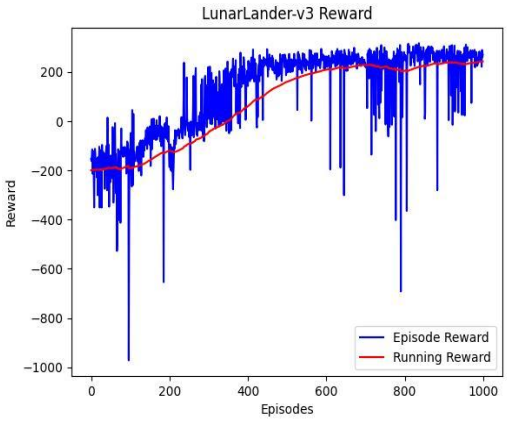
Input for NoisyDueling_DQN for Stochastic CartPole-v1



NoisyDueling_DQN for Stochastic CartPole-v1

```
Enter a seed value (e.g., 42): 10
Choose mode: Deterministic (D) or Stochastic (S): d
Running in Deterministic mode
Do you want to use the HRL agent? (Y/N): y
```

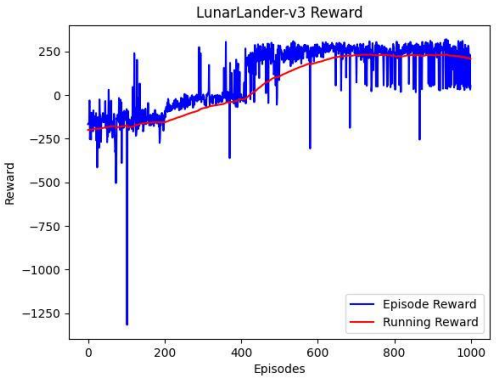
Input used for HRL_NoisyDueling_DQN for Deterministic LunarLander-v3



HRL_NoisyDueling_DQN for Deterministic LunarLander-v3

```
Enter a seed value (e.g., 42): 10
Choose mode: Deterministic (D) or Stochastic (S): s
Running in Stochastic mode
Enter gravity noise (e.g., 0.1 for 10% increase): 0.1
Enter wind power (e.g., 15.0): 1
Enter turbulence power (e.g., 1.5): 1
Do you want to use the HRL agent? (Y/N): y
```

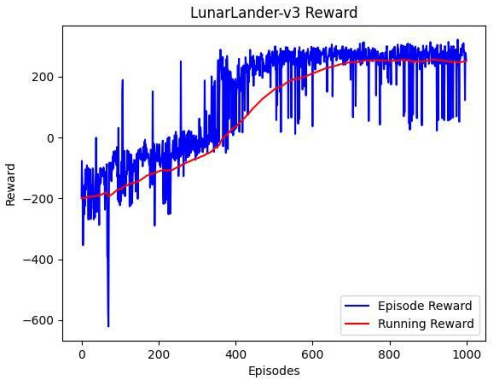
Input used for HRL_NoisyDueling_DQN for Stochastic LunarLander-v3



HRL_NoisyDueling_DQN for Stochastic LunarLander-v3

```
Enter a seed value (e.g., 42): 10
Choose mode: Deterministic (D) or Stochastic (S): d
Running in Deterministic mode
Do you want to use the HRL agent? (Y/N): n
```

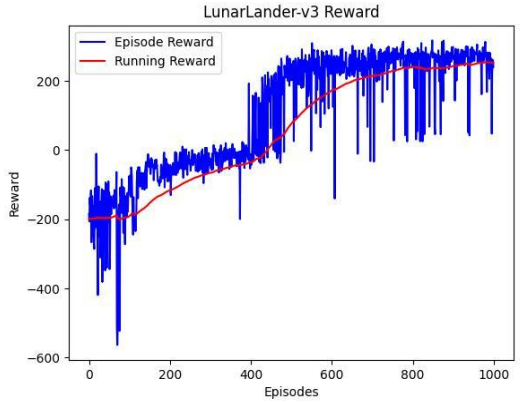
Input used for NoisyDueling_DQN for Deterministic LunarLander-v3



NoisyDueling_DQN for Deterministic LunarLander-v3

```
Enter a seed value (e.g., 42): 10
Choose mode: Deterministic (D) or Stochastic (S): s
Running in Stochastic mode
Enter gravity noise (e.g., 0.1 for 10% increase): 0.1
Enter wind power (e.g., 15.0): 1
Enter turbulence power (e.g., 1.5): 1
Do you want to use the HRL agent? (Y/N): n
```

Input used for NoisyDueling_DQN for Stochastic LunarLander-v3



NoisyDueling_DQN for Stochastic LunarLander-v3