

ID6002WReinforcement Learning Project Report

short line

Soumya Mukherjee  
CH24M571

M.Tech Industrial AI

# Introduction

The RL project asks us to solve a multi-product inventory management problem. The objective is to develop an ordering policy that minimizes total operational costs within a simulated warehouse environment. The problem setup , algorithmic setup and key observations shall be discussed herein within .

**Problem Setup**

We have 3 distinct products . The goal is to minimize the cumulative cost over a 50-day simulation period, or "episode". The agent's performance is measured by its ability to balance various cost factors through intelligent daily ordering decisions.

We have the *InventoryEnv*  class provided to us . Conditions of the environment are:

* **Warehouse Capacity:** The total volume of all products stored cannot exceed **1000 cubic units**. If arriving orders cause the inventory to exceed this limit, the excess items are discarded, incurring a penalty.
* **Product Specifications:**
  1. **Volumes:** Product 1: 2.0, Product 2: 3.0, Product 3: 1.5 volume units/item.
  2. **Initial Inventory:** 100 units for each of the three products.
  3. **Lead Times:** Orders for Product 1, 2, and 3 arrive in 3, 2, and 1 day(s), respectively.
* **Cost Structure:** The total cost is an aggregate of four components:
  1. **Holding Cost:** A charge of **$5.00 per unit of volume** per day for items in storage.
  2. **Stockout Cost:** A penalty for unfulfilled demand, set at **₹400.00** for Product 1, **₹500.00** for Product 2, and **₹300.00** for Product 3.
  3. **Ordering Cost:** A fixed cost incurred per product when an order is placed, regardless of quantity. The costs are  
      **₹80.00** for Product 1, **₹200.00** for Product 2, and **₹120.00** for Product 3.
  4. **Discard Cost:** A penalty for discarding items due to warehouse over-capacity, set at **₹200.00** for Product 1, **₹250.00** for Product 2, and **₹150.00** for Product 3.
* **Demand:** During training, daily demand for each product is stochastic, generated from a Poisson distribution with mean values of 30, 25, and 35, respectively. For final evaluation, the agent is tested against fixed, deterministic demand sequences.

# Algorithmic approach :

We had started with the simple Deep Q Networks architecture . While a good starting point, DQN is known to sometimes overestimate Q-values, which can lead to suboptimal policies.

To address this, we moved to the Dueling Double Deep Q Network (D3QN) architecture. This architecture combines two key advancements over standard DQN:

Double DQN: This technique helps to mitigate the overestimation bias by decoupling the action selection and the target Q-value calculation. Instead of using the same network to select the next action and evaluate its Q-value, Double DQN uses the online network to select the action and the target network to evaluate the Q-value of that action.

Dueling Networks: This architecture splits the Q-network into two streams: one that estimates the value of the state and another that estimates the advantage of each action. The total Q-value is then a combination of these two streams. This allows the network to learn the value of a state independently of the actions, which can be particularly useful in states where the choice of action doesn't significantly impact the outcome. In our inventory problem, this could help the agent understand the inherent value of a certain inventory level regardless of the immediate ordering decision.

By combining these two techniques, D3QN provides a more stable and effective learning process, especially in environments with large action spaces and complex reward landscapes like our inventory management simulation.

Complexity of the Problem: The large action space (1331 actions) and the delayed effects of ordering (due to lead times) make this a challenging RL problem. The D3QN architecture is a strong choice for handling these complexities.

Hyperparameter Tuning: The performance of the agent is highly sensitive to hyperparameters like the learning rate (lr), discount factor (gamma), and the epsilon decay rate. The chosen values (lr=1e-3, gamma=0.99, epsilon\_decay=0.995) are reasonable starting points, but extensive tuning would likely be required to achieve optimal performance and a lower average cost on the leaderboard.

Potential for Reward Shaping: The project description suggests using reward shaping to accelerate learning. Implementing a bonus for maintaining a balanced inventory or a penalty for erratic ordering could guide the agent to a more effective policy faster than relying solely on the sparse and often negative base reward signal.

Inferences made :

D3QN was essential for handling the complex action space and delayed rewards better than basic DQN.

The agent must learn to anticipate future needs due to lead times, not just react to the current state.

Optimal policy requires balancing holding, stockout, ordering, and discard costs.

Reward shaping helps guide the agent towards desirable behaviors like balanced inventory and stable ordering.

Prioritized Experience Replay is used because some experiences are more valuable for learning than others.

State normalization is important for the neural network to learn effectively with different state feature scales.

Lessons learnt :

Selecting the Right Algorithm: Simple DQN had limitations (overestimation). Dueling Double DQN (D3QN) was chosen for better stability and Q-value estimation in complex environments with large action spaces and delayed rewards.

Importance of Hyperparameters: Agent performance is highly sensitive to hyperparameters. Tuning is crucial for effective training.

Benefit of Reward Shaping: Guiding the agent with shaped rewards accelerates learning in environments with sparse feedback.

Value of Experience Replay: Prioritized Experience Replay improves learning efficiency by focusing on impactful experiences.

Monitoring is Key: Tracking training progress and evaluating performance is essential to gauge learning effectiveness. We have ran for 500 epochs , it could be ran for much more epochs to have a better reward score .