

Reinforcement Learning  
Group Project

Lecturer - Stergios Christodoulidis

**Minimising Food Waste and   
Maximising Profit in a Grocery Shop**



**Javier Armando Gonzalez Lizcano**

**Jean Nicolas Hofmann**

**Elliot Massey**

# Abstract

Our group reinforcement learning project aims to address the dual challenges of minimising food waste and maximising profit in a grocery shop environment using reinforcement learning techniques. Through simulation, we have developed and tested a set of intelligent agents capable of making real-time decisions to optimise inventory management. Our findings demonstrate significant reductions in food waste and substantial increases in profit, highlighting the potential of reinforcement learning in improving operational efficiency and sustainability within the retail sector.

# Motivation

Grocery shops play a central role in our lives and society as nearly everybody needs to visit them on a regular basis. Consequently, even minor changes in their processes have a large impact on society due to their magnitude. Their importance also explains why grocery shops are faced with intense competition and the need to drive sales, often prioritising ensuring product availability. This emphasis on maintaining high levels of on-shelf availability and offering a wide variety of products is centred towards enhancing customer satisfaction. However, this approach inevitably leads to a significant surplus of inventory, which if not sold before its expiration date, contributes to the problem of food waste.

In general, over 30% of food is lost or wasted each year (FAO1). Given the large number of hungry people in the world, food waste is not only inefficient, it’s a social justice issue. Additionally, food waste has an enormous environmental impact as it is a huge source of greenhouse gas emissions. Due to the wasted natural resources, food waste is responsible for around 10% of worldwide carbon emissions (WWF2). Considering the magnitude of small changes in processes of grocery stores, even small reductions in food waste created in the stores could have a major impact on reducing carbon emissions and contribute to the fight against world hunger. However, since grocery stores act in a competitive environment, profit maximisation also needs to be considered.

# Environments

In order to experiment between real-world environments (grocery shop) and reinforcement learning agents (DQN), simulation environments seem to be a feasible way to do so ([Cestero et al., 2022](#_heading=h.4d34og8)). Our grocery shop environment replicates the complexities of a real-world retail setting and allows for dynamic interactions between customers and products.

***State Space*** - In our project, the state space consists of tuples representing the quantity of products in stock and the closest time remaining until expiration. The size of this state space is determined by the capacity of the grocery shop and the expiration time of products. As we know, the expiration time can vary a lot between products - there is fresh produce such as bread and fresh fruit that have an incredibly short shelf life compared to tinned goods which can last for years unopened. In our present work, the simulation is done for one product, where at each time step there is a demand governed by a poisson distribution with parameter .

***Action Space*** - Actions in our project are the quantity to restock at each time space, which depending on the setup of the environment, ranges from zero to max\_stock capacity.

***Rewards*** - At every time step, the reward received by the agent has four components:

| ***Utility*** | ***Waste Penalty*** |
| --- | --- |
| ***Maintenance Cost*** | ***Unavailability Penalty*** |

***d***: Demand at the time step, it is a sample from a poisson distribution with parameter .  
***AS***: Actual stock ***SP*** : Selling price ***C*** : Cost of product ***WC***: Waste cost

***UW***: Units wasted ***MC***: Maintenance cost ***DNC***: Demand not covered

***US***: Units in stock

**Total reward = Utility - Waste Penalty - Maintenance Cost - Unavailability penalty**

The configurable parameters are: SP, C, WC, MC, and parameters of the demand distribution.

Time step procedures:

* ***Stock*** - If the action taken is to re-stock n units of a product, the actual stock quantity increases, and the track of the expiration time of these new units are logged in a list of dictionaries (***expiration\_data***) of the environment.
* ***Units wasted*** - Each expiration\_data item is updated by decreasing the expiration time by one. The UW (Wasted units) at every time step are those where the expiration time is less than zero.
* ***Demand not covered*** - At each time step, if the demand is not covered by the actual stock and the action (re-stock quantity), a ***penalty*** is put in place.

# 

# Agents

The goal of our [agents](#_heading=h.17dp8vu) is to maximise overall profit, instead of simply sales. This means that the cost of waste and missed sales are both taken into account.

***Base Agent*** - A base agent is used as a baseline to compare the performance of different policies, and other agents. The base agent policy is as follows: if the quantity of a product falls below half of its maximum capacity or its expiration time is nearing, the agent takes action to restock up to the maximum capacity , otherwise, it refrains from restocking to avoid unnecessary surplus i.e. the restock is zero.

***Temporal Difference (TD) Agents***

1. **Off-Policy Learning using Q-learning:** The Q-learning agent selects actions based on the maximum estimated Q-value for each state-action pair and updates its Q-values using the temporal difference error between observed and predicted rewards. The agent balances exploration and exploitation through an epsilon-greedy policy, where it chooses random actions with a probability epsilon and selects the action with the highest Q-value otherwise.
2. **On-Policy Learning using Expected Sarsa:** Expected Sarsa is an on-policy temporal difference learning algorithm that estimates action values using the expected value of the next state-action pair under the current policy. This approach allows the agent to learn directly from its actions while considering the expected future rewards, leading to more stable learning.

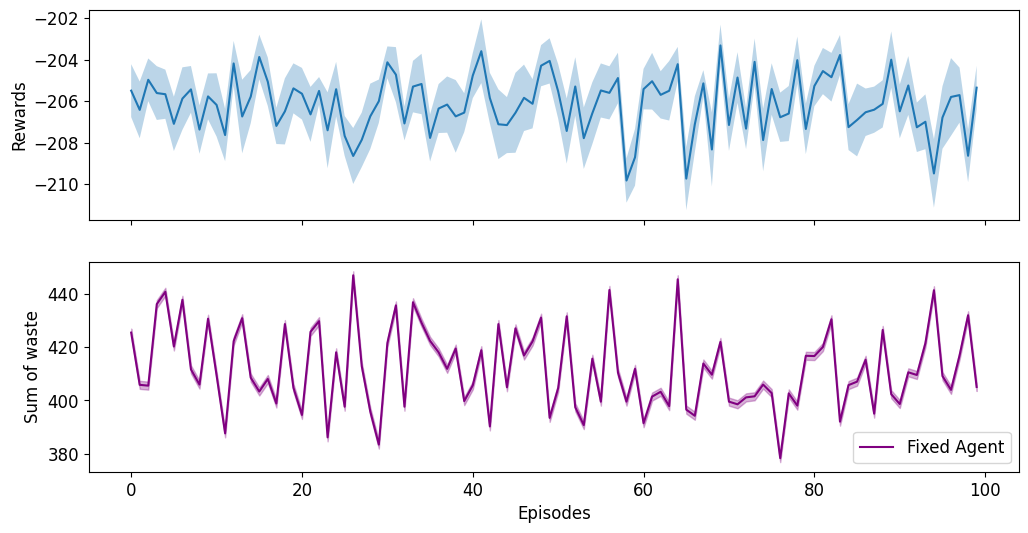
***Deep Q-Network (DQN) Agent***  - The DQN agent employs a deep neural network, specifically a fully connected architecture, to approximate the Q-value function. By leveraging neural networks, the agent is capable of learning complex patterns and making informed decisions based on the observed states. Through training, the DQN agent adapts its policy function to optimise inventory restocking decisions, ultimately aiming to maximise profit while minimising waste and missed sales. The agent was able to successfully adjust the restock decision by adjusting the policy function with a fully connected neural network.

# Experiment Setup

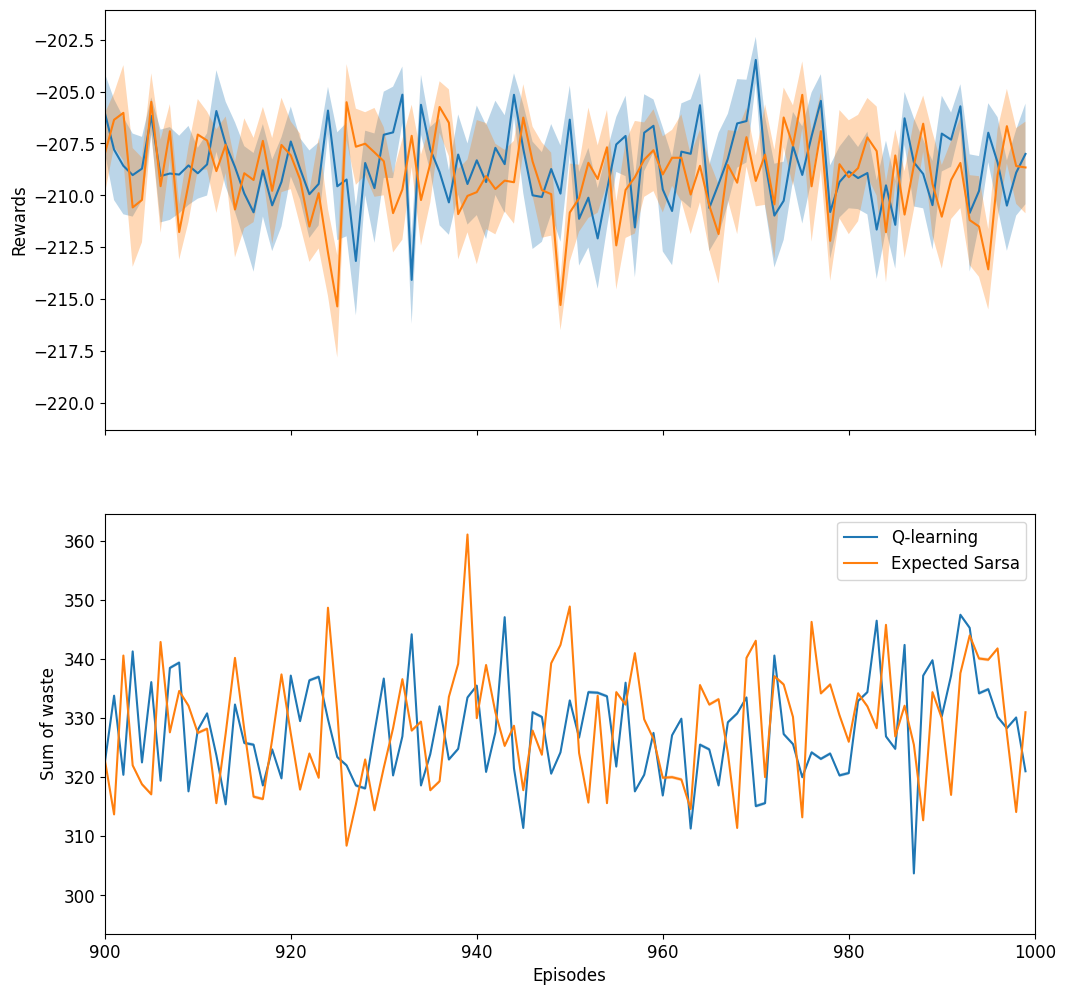
The simulation environment is configured to mimic a typical grocery shop setting, with parameters such as initial stock capacity, maintenance costs, expiration times, buying and selling prices, weekly demand, and maximum time horizon specified.

| Key features of the environment include | |
| --- | --- |
| **Capacity**: Maximum stock the shop can hold. | **Buying Price**: Price at which products are purchased. |
| **Selling Price**: Price at which products are sold. | **Weekly Demand**: Expected demand for products per week. |
| **Expiration Time**: Shelf life of products in days. | **Max Time**: Maximum time horizon for the simulation. |
| **Gamma**: Discount factor for future rewards. | **Expiration Cost**: Cost associated with expired products. |
| **Max Loss**: Maximum acceptable loss for the shop. | **Maintenance Cost**: Cost associated with maintaining the inventory. |

# Results

***Baseline*** - The Baseline enables us to compare results of the experiments with other agents and policies. In this case, we can see that the rewards vary around -206 and the sum of waste around 420. However, both seem to vary strongly between different episodes.   


***Temporal Difference*** - The experiments showed that this type of agent doesn’t perform well in adjusting inventory levels, and responding to customer behaviour in real-time. In our experiments, the average reward after 1000 episodes is still negative, and around the same value as the baseline agent. In comparison to the baseline results, the sum of waste seems to be slightly improved to a smaller amount of around 335.



***DQN Agent*** - From these experiments we can see that the DQN Agent does well over the episodes in improving mean waste and mean reward. As shown in the graph, after episode 400 the mean waste is around 50 and the mean reward around 1700, which is well below / above the baseline results.



***Policy Evaluation*** - We decided to test our learned policy under different conditions to assess its robustness and adaptability. By varying parameters such as the cost of waste, selling price, and demand, we can observe how well the policy performs across a range of scenarios:

[*Impact of cost of waste*](#_heading=h.26in1rg)*:*When we vary the penalty on the cost of waste in the environment, and we use the same policy learned by the DQN agent. A higher waste cost induces less maximum total waste per episode.

[*Impact of product selling price*](#_heading=h.35nkun2)*:*When the selling price is changed in a range between 1 to 4 units, using the same policy learned by the DQN agent. We can observe that as the selling price increases the profits grow larger but so does the sum of waste. However, the speed at which the waste reduces to its low level is increased as selling price increases.

[*Impact of change in demand*](#_heading=h.lnxbz9)*:*Our experiments show the influential impact the demand parameter has on both reward and waste. Our agent was trained on a demand value equal to 5. Below this value, the reward obtained is negative, while the waste is around 190 units. For a demand double the trained value, the average waste decreases dramatically, however, its maximum value increases almost double. It is interesting to observe how a higher demand (20) decreases the waste (as expected), but under the trained policy, the profit goes to zero, this is most likely due to the cost of maintenance of products in stock. **In general,** the results we obtained demonstrate the effectiveness of our reinforcement learning approach in minimising food waste and maximising profit. Compared to the baseline, our agents achieved a **[90%]** reduction in food waste and a **[800%]** increase in profits for the simulated grocery shop environment.

# Conclusion

Our project highlights the potential of reinforcement learning as a powerful tool for addressing food waste and sales optimisation challenges. While our findings are promising, several limitations and areas for future research exist, for example, including multi-product dynamics, the impact of cannibalisation between products, seasonality, and others.

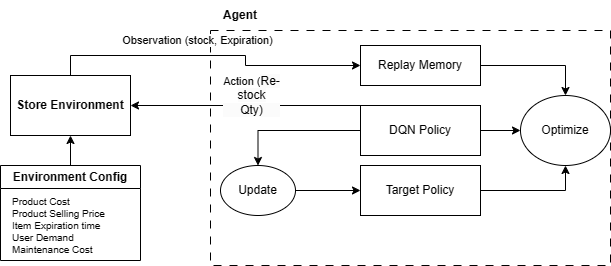
This is just one approach to fight the problem of food waste. Another promising approach for example is the app "Too Good To Go" - users can purchase “surprise bags" or specific items at discounted prices, that would otherwise be discarded from local stores. An idea that helps individuals access affordable meals but also supports businesses in minimising food waste. A community-driven approach to combating food waste at scale.

# Bibliography

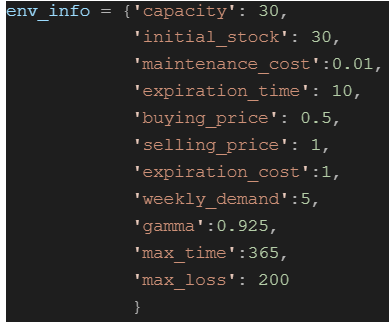
* Julen Cestero, Marco Quartulli, Alberto Maria Metelli, and Marcello Restelli. Storehouse: a reinforcement learning environment for optimising warehouse management. In International Joint Conference on Neural Networks, pp. 1–9. IEEE, 2022.

# Appendix

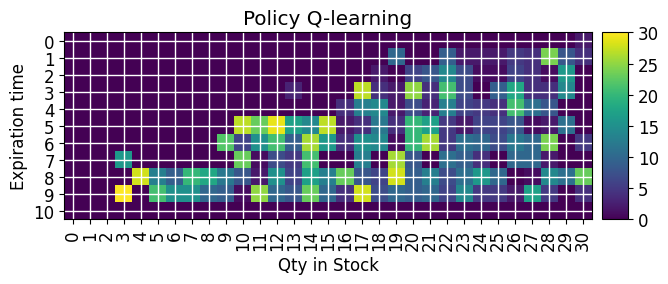
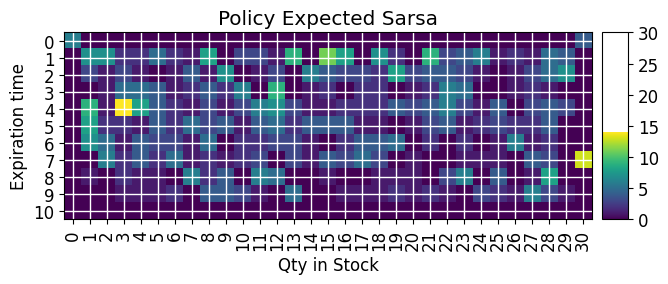
### Agent Environment Setup

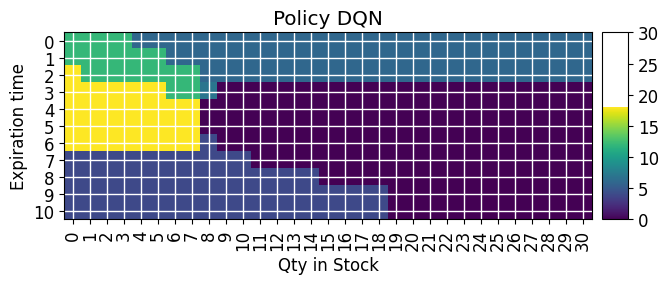


### Store Environment Setup



**Q-Learning v.s. Expected Sarsa v.s. DQN Policies**





### Change in Cost of Waste

|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
|  |  |

### Change in Demand

### Change in Selling Price

|  |  |
| --- | --- |
|  |  |