Deep Reinforcement Learning for Autonomous Portfolio Management: A Policy Gradient Approach

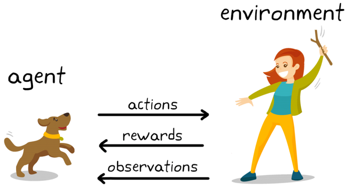
Jacopo Signó

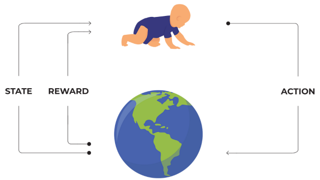
April 2024

What is Reinforcment Learning?

The Reinforcement Learning paradigm was inspired **by the way living beings learn**.

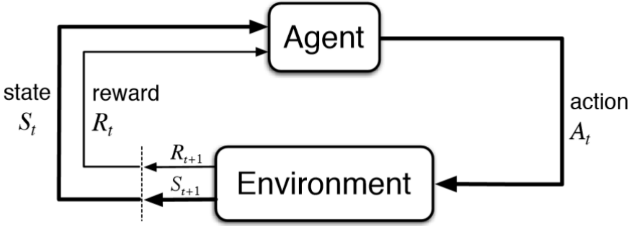
Biological brains interpret signals such as pain as negative reinforcements and pleasure as positive reinforcements.





What is Reinforcment Learning?

Reinforcement learning is a machine learning paradigm where an agent tries to learn the sequence of optimal decisions to take to maximise the obtained reward over time by interacting with an environment.



$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

What is Reinforcment Learning?

- **Policy π:** defines the strategy with which the agent makes decisions.

- **Value Function**: represents the expected long-term return for a given state following a policy π

$$v_\pi(s) = \mathbb{E}_\pi \left[G_t \mid S_t = s \right]$$

The Goal

The goal of our agent is to dynamically re-allocate assets in a portfolio to maximize his returns.

The Goal

- Number of stocks: M
- Portfolio vector:

$$\mathbf{w}_t = [w_{1,t}, w_{2,t}, ..., w_{M,t}] \ and \ \sum_{i=1}^{M} w_{i,t} = 1$$

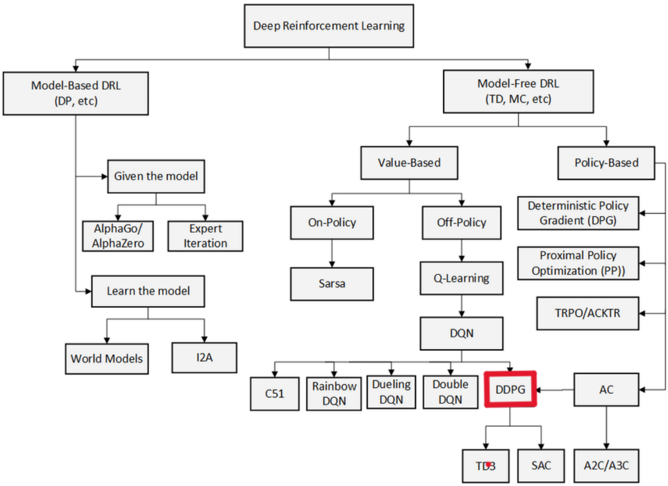
with $w_{i,t} \in [0, 1]$

- Goal: maximize $E[V_T]$ over a period of time T

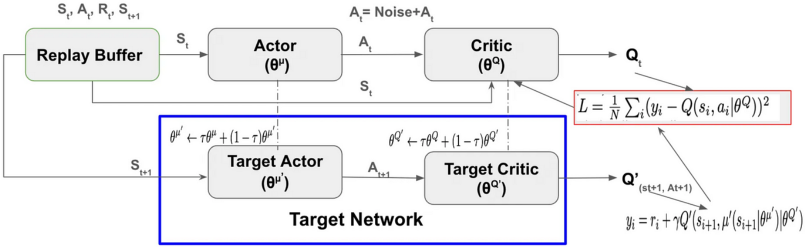
Reinforcement Learning Components in Asset Allocation Optimization

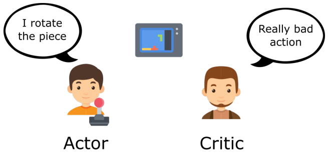
- **Agent:** The investor (software portfolio manager)
- **Environment:** The financial market.
- **State:** stocks' information that the agent has available to make his decisions.
- **Action**: $\mathbf{a}_t = \mathbf{w}_{t+1}$ with $w_{i,t+1} \in [0, 1]$
- **Reward:** $r_t = (V_t - V_{t-1}) - (SP500_t - SP500_{t-1})$

Understanding Deep Deterministic Policy Gradient (DDPG)



Understanding Deep Deterministic Policy Gradient (DDPG)





DDPG

Algorithm 1 DDPG algorithm

Randomly initialize critic network $Q(s,a|\theta^Q)$ and actor $\mu(s|\theta^\mu)$ with weights θ^Q and θ^μ.
Initialize target network Q' and μ' with weights $\theta^{Q'} \leftarrow \theta^Q$, $\theta^{\mu'} \leftarrow \theta^\mu$
Initialize replay buffer R
for episode = 1, M **do**
  Initialize a random process $\mathcal{N}$ for action exploration
  Receive initial observation state s_1
  for t = 1, T **do**
    Select action $a_t = \mu(s_t|\theta^\mu) + \mathcal{N}_t$ according to the current policy and exploration noise
    Execute action a_t and observe reward r_t and observe new state s_{t+1}
    Store transition (s_t, a_t, r_t, s_{t+1}) in R
    Sample a random minibatch of N transitions (s_i, a_i, r_i, s_{i+1}) from R
    Set $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$
    Update critic by minimizing the loss: $L = \frac{1}{N}\sum_i (y_i - Q(s_i, a_i|\theta^Q))^2$
    Update the actor policy using the sampled policy gradient:

$$\nabla_{\theta^\mu} J \approx \frac{1}{N}\sum_i \nabla_a Q(s,a|\theta^Q)|_{s=s_i,a=\mu(s_i)} \nabla_{\theta^\mu}\mu(s|\theta^\mu)|_{s_i}$$

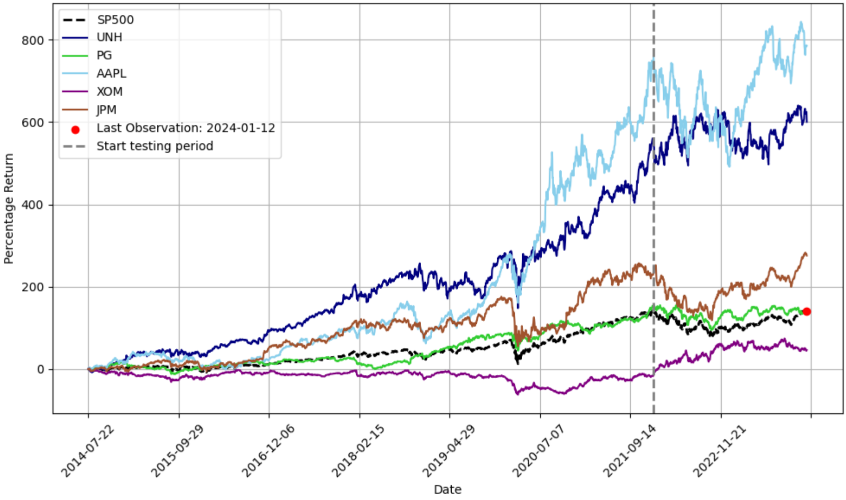
    Update the target networks:

$$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}$$

$$\theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}$$

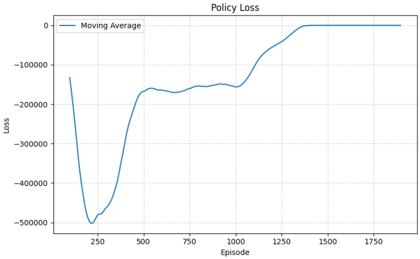
  end for
end for

Data

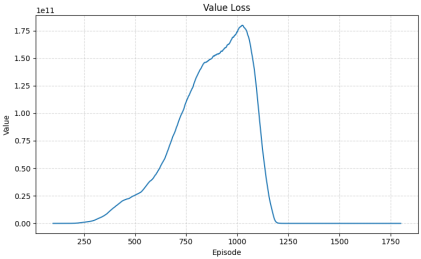

SP500
UNH
PG
AAPL
XOM
JPM
Last Observation: 2024-01-12
Start testing period
Percentage Return
800
600
400
200
0
2014-07-22
2015-09-29
2016-12-06
2018-02-15
2019-04-29
2020-07-07
2021-09-14
2022-11-21
Date

Training Phase

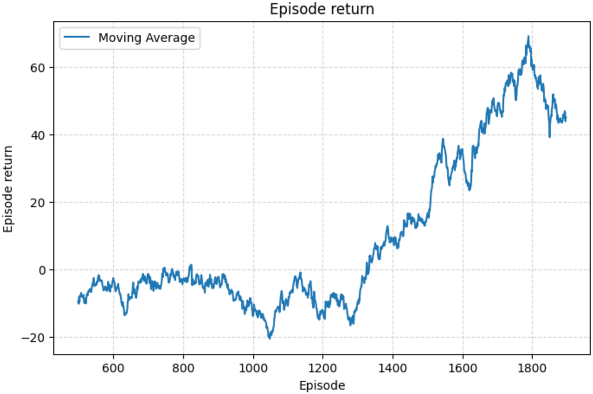
Policy Loss



Value Loss

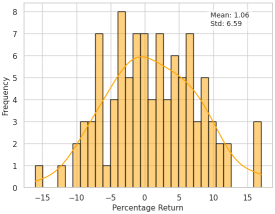


Training phase



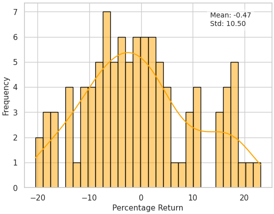
Testing Phase

1 month



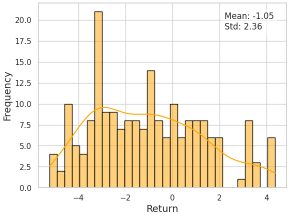
Mean = 1.06
Std = 6.59

6 months



Mean = -0.47
Std = 10.50

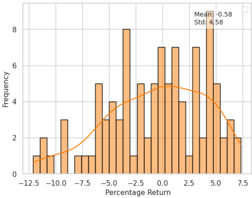
1 year



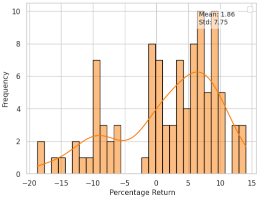
Mean = -1.05
Std = 2.36

Benchmark returns vs Portfolio returns

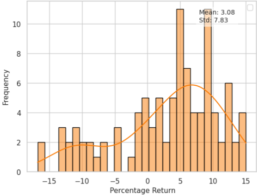
1 month	6 months	1 year



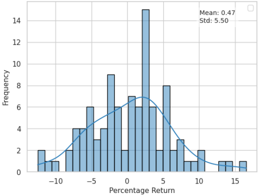
mean = -0.58 std = 4.58



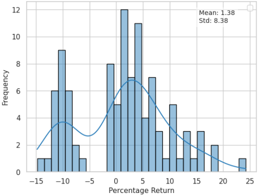
mean = 1.86 std = 7.75



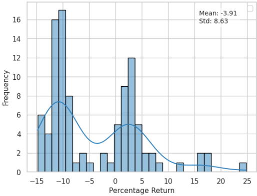
mean = 3.08 std = 7.83



mean = 0.47 std = 5.50



mean = 1.38 std = 8.38



mean = -3.91 std = 8.63

Conclusion

A fundamental assumption underlying the model is that the state representation respects the **Markov property**.

We can therefore conclude that the state we proposed is not Markov and does not allow us to implement an agent that is capable of managing a portfolio in an optimal manner

Conclusion

Thank you very much for your attention!