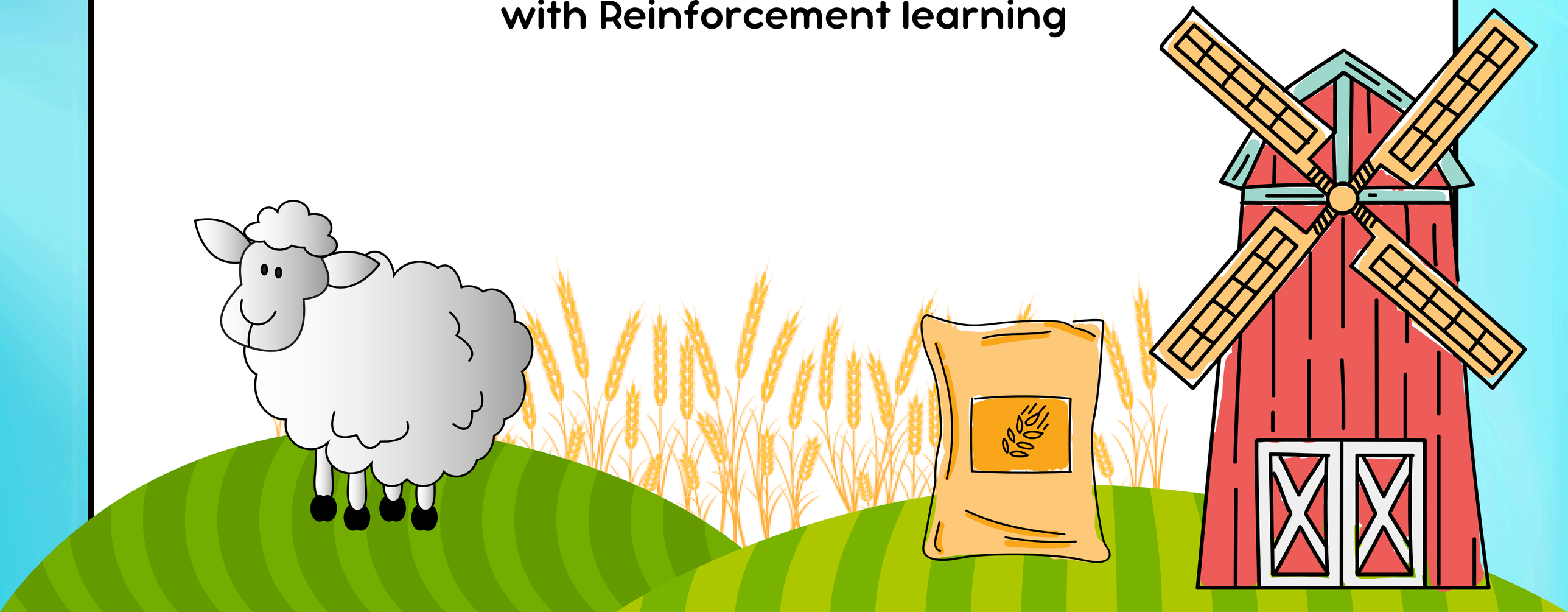
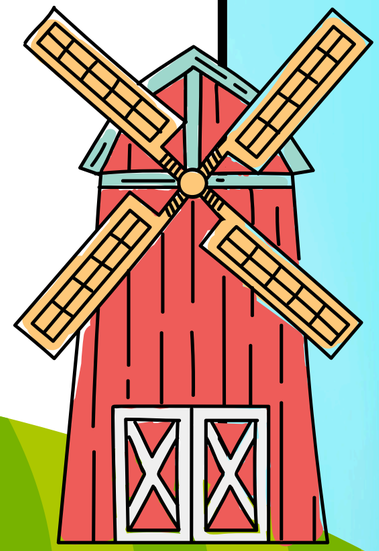
Manage a farm

with Reinforcement learning



The environment

- Initial budget: € 2000
- Cost of buying a sheep: € 1000
- Cost of growing wheat: € 20
- Wool selling price: € 10
- Weath selling price: € 50
- Fixed cost for processing wool: € 9
- Storm probability: 30 %
- Dynamic probability of reproduction : $p = \left(\frac{n_{sheeps\_bought}}{n_{sheeps}} \right)^{Penalty}$



The environment

Actions available

Buy a sheep

€ −1000

100 %: **Sell wool for [€ 10 · sheeps] − € 9**

p: **X ~ B(n, p) new sheeps**

Grow wheat

€ −20

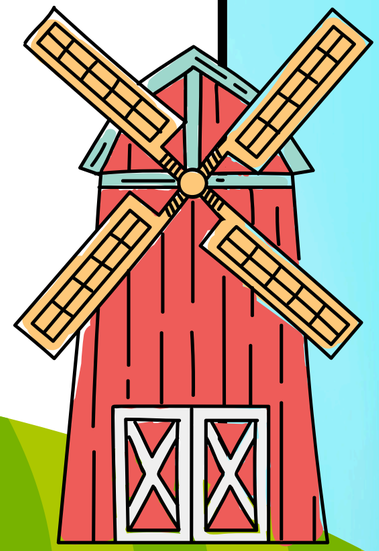
70 %: **Wheat harvest yields € 50** + **Sell wool for [€ 10 · sheeps] − € 9**

30 %: **A storm occurs € 0**

Do not invest

€ 0

Sell wool for [€ 10 · sheeps] − € 9



The environment

Reward

r = Δ · year

r = Budget

r = Δ ^ year

r = Δ

Delta budget

999

Δ

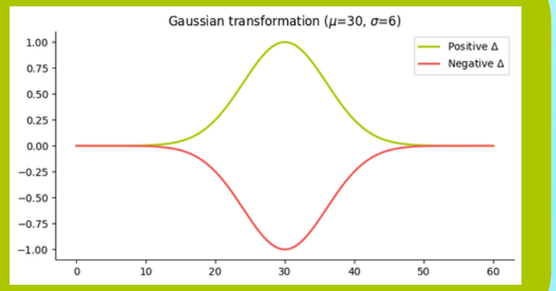
-999

Normalization

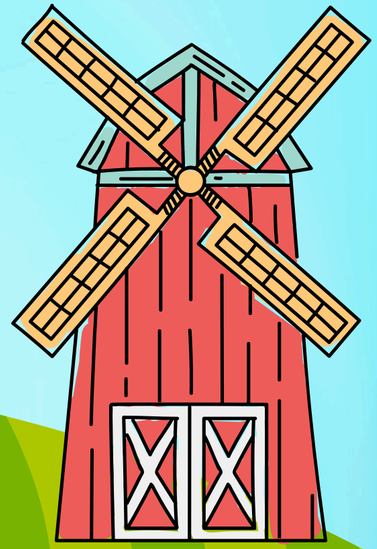
1

-1

Gaussian transformation

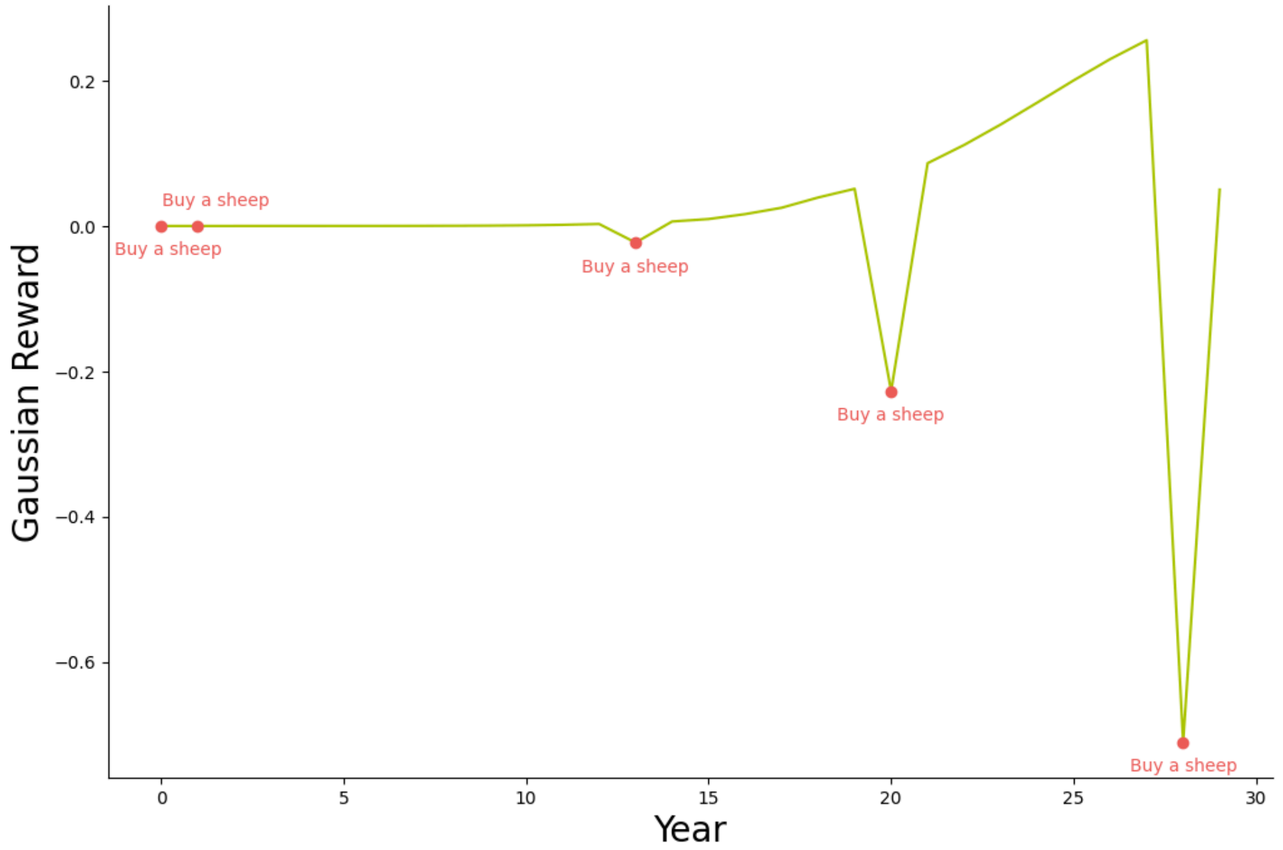


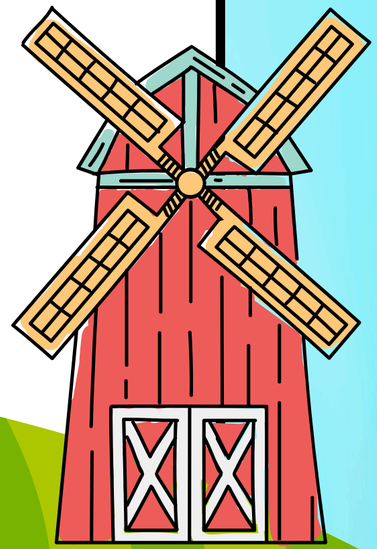
r



The environment

Reward

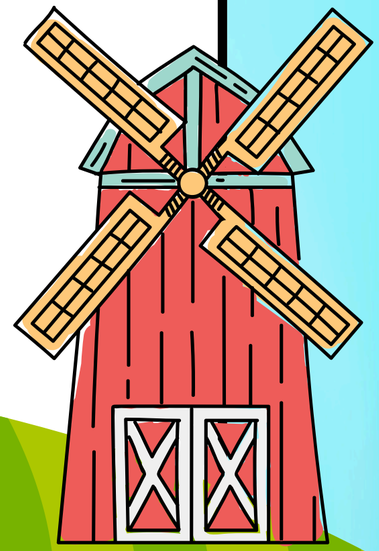




Features selection

state = X(s,a) =

	Budget	Sheeps bought	Total sheeps	Bought sheep ratio	Years
Buy sheep					
Grow wheat					
Do not invest					



Agent selection

Value Function Approximation

Linear approximation of the state-value function to overcome the limitation of having a continuos state space

$$\hat{Q}(s, a, \mathbf{w}) = \mathbf{x}(s, a)^T \mathbf{w}$$

Monte Carlo

$G_t(s_t, a_t) = \sum_{i=t}^{m} \gamma^i r_{e,i}$ where $e = \langle s_{e,1}, \pi(s_{e,1}), r_{e,1}, s_{e,2}, \ldots, s_{e,m} \rangle$

$$\Delta(\mathbf{w}) = \alpha \left(G_t(s, a) - \hat{Q}(s_t, a_t, \mathbf{w}) \right) \nabla \mathbf{w} \hat{Q}(s_t, a_t, \mathbf{w})$$

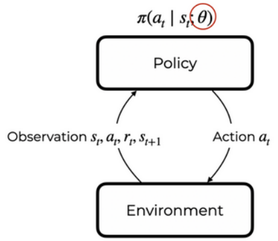
SARSA

$$\Delta(\mathbf{w}) = \alpha \left[r_{t+1} + \gamma \hat{Q}(s_{t+1}, a_{t+1}, \mathbf{w}) - \hat{Q}(s_t, a_t, \mathbf{w}) \right] \nabla_{\mathbf{w}} \hat{Q}(s_t, a_t, \mathbf{w})$$

Policy gradient methods

Update of a parametrized policy to maximize the expected total return

$$\nabla_\theta \mathbb{E}_{\pi_\theta} G(\tau) = \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t | s_t) G(\tau)$$



REINFORCE with Advantage function

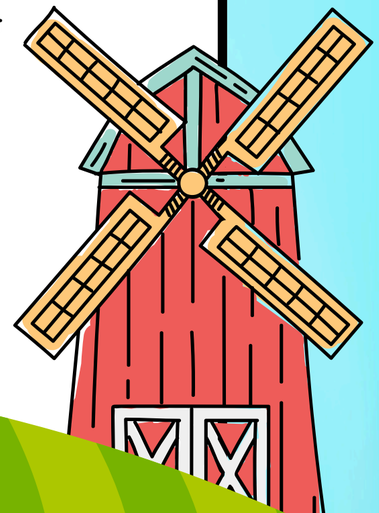
Decompose the state-value into a Policy and a Value component

$$A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s)$$

Update two sets of parameters

$\mathbf{w} \leftarrow \mathbf{w} + \alpha^{\mathbf{w}} \delta \nabla_{\mathbf{w}} V(s_t; \mathbf{w})$ Value function weights

$\theta \leftarrow \theta + \alpha^\theta \gamma^t \delta \nabla_\theta \log \pi(a_t \mid s_t; \theta)$ Policy function weights



Agent selection

Montecarlo VFA

Good state generalization but difficulty understanding when to stop buying sheeps

↓

Training succesful but loss of wealth

SARSA VFA

Instability during training (exploding gradient)

↓

Training failed

REINFORCE with Neural Advantage

Optimal state generalization and training stability

↓

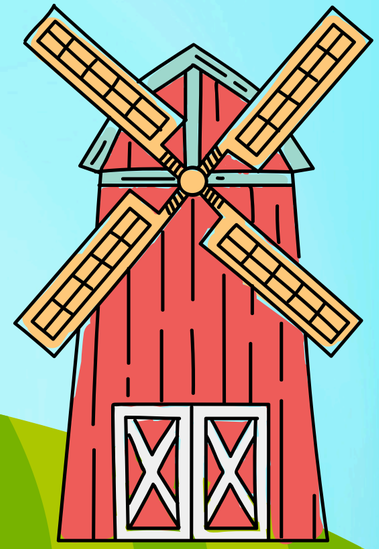
Best positve results

REINFORCE with Advantage

Training stability but sub-optimal approximation

↓

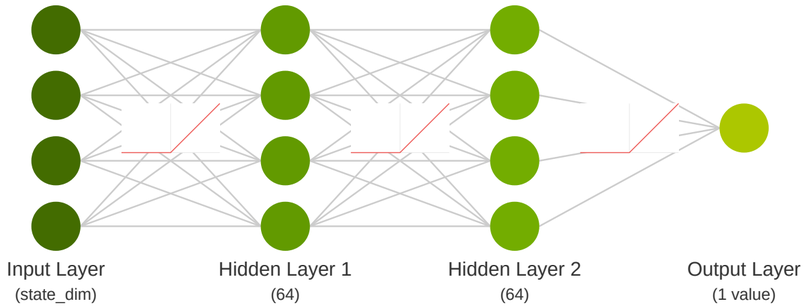
Positive but non-optimal results



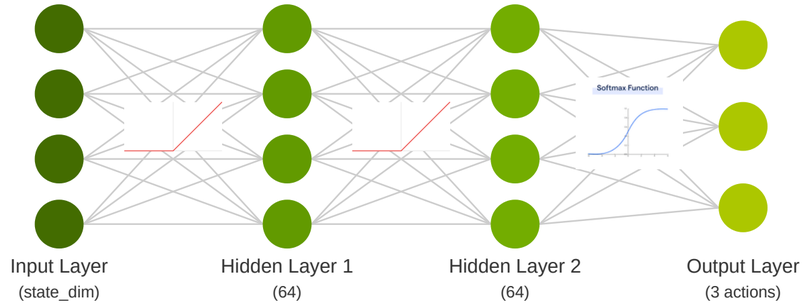
REINFORCE with Advantage

Architectures

Value Network

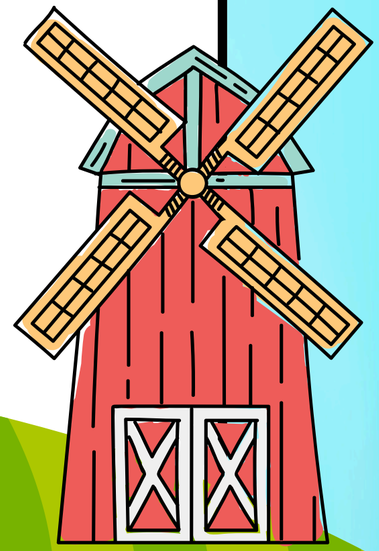


Policy Network



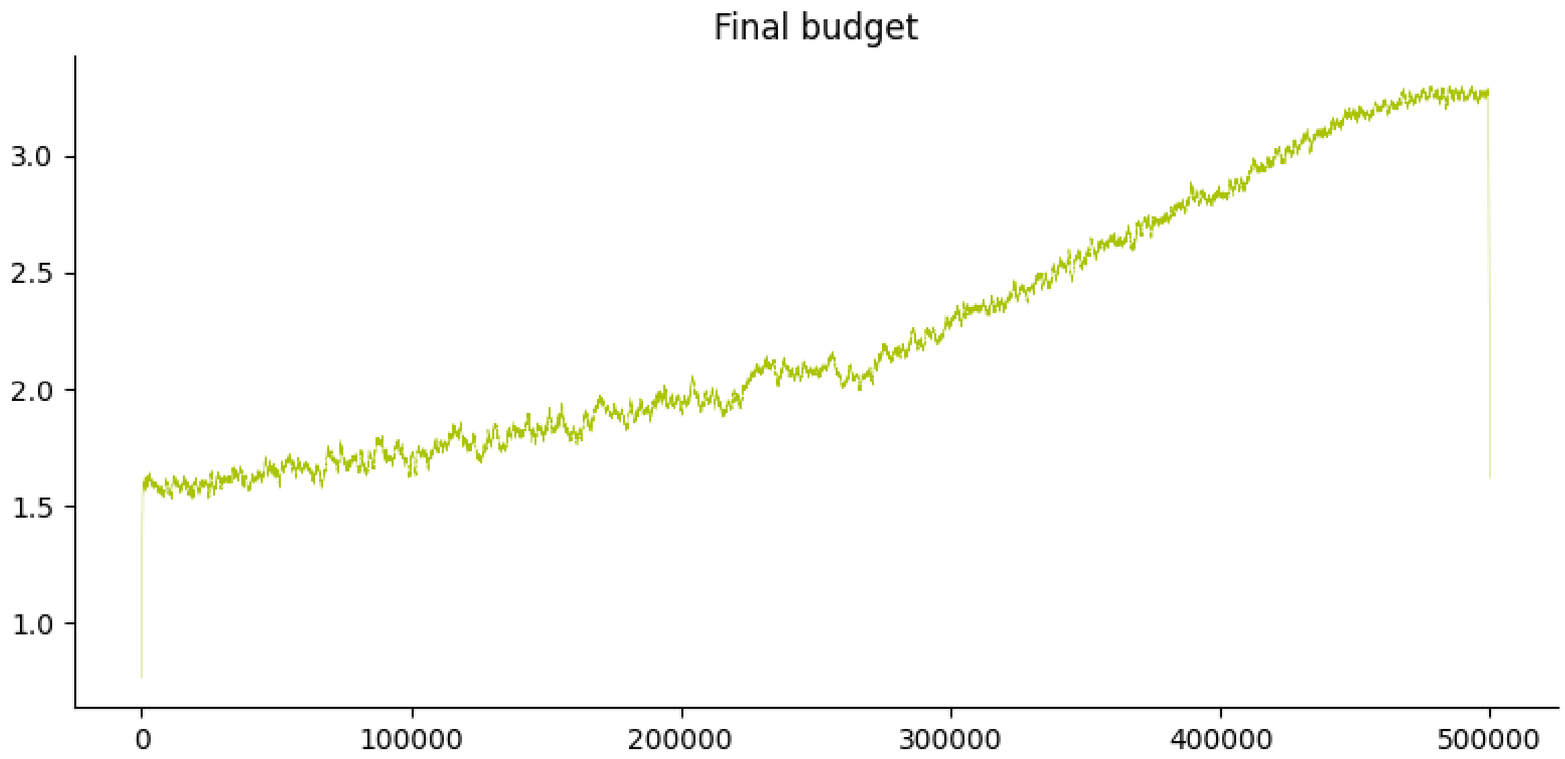
Main strenghts

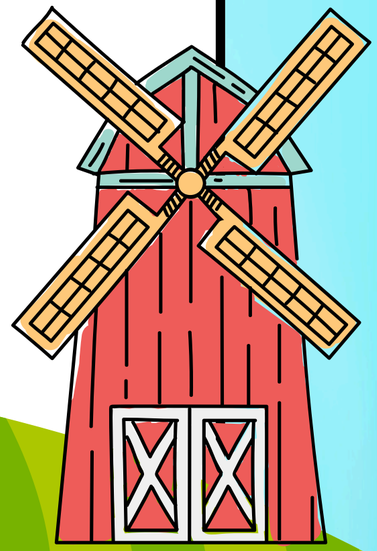
- Policy is not deterministic, so that there is always some exploration
- Ability to generalize in continuous state spaces
- Better approximation of the value functions thanks to Neural networks



Results

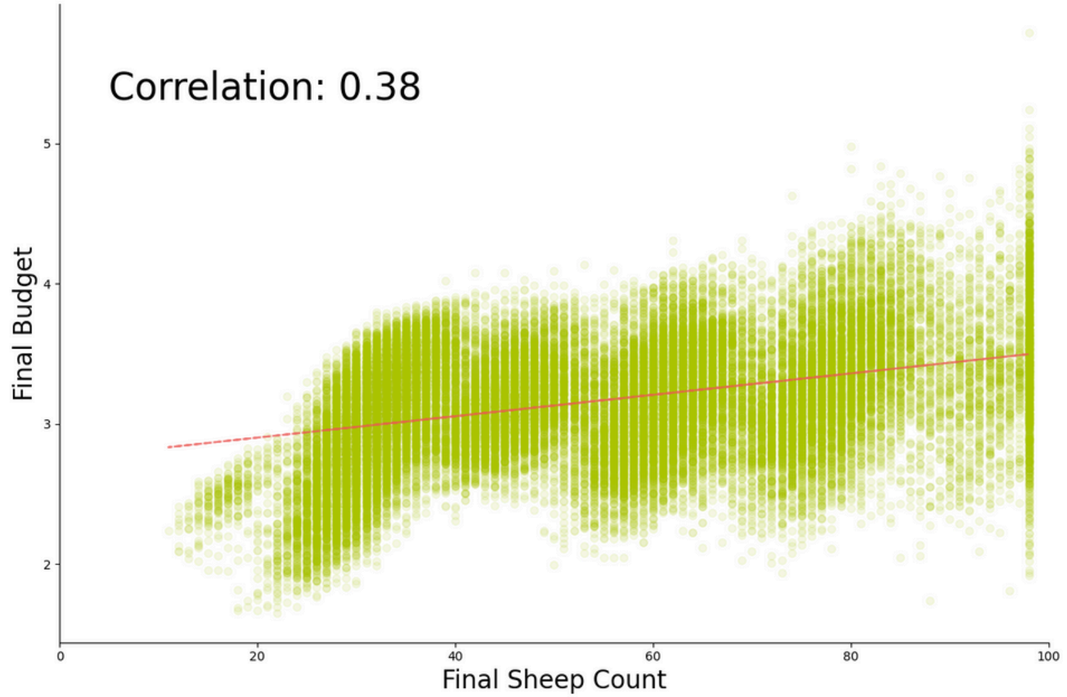
Final budget during training

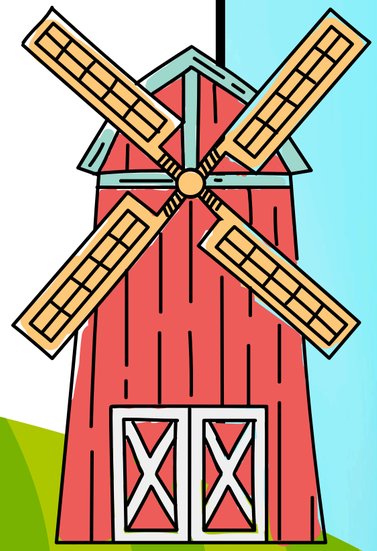




Results

Correlation: Sheeps vs Final budget

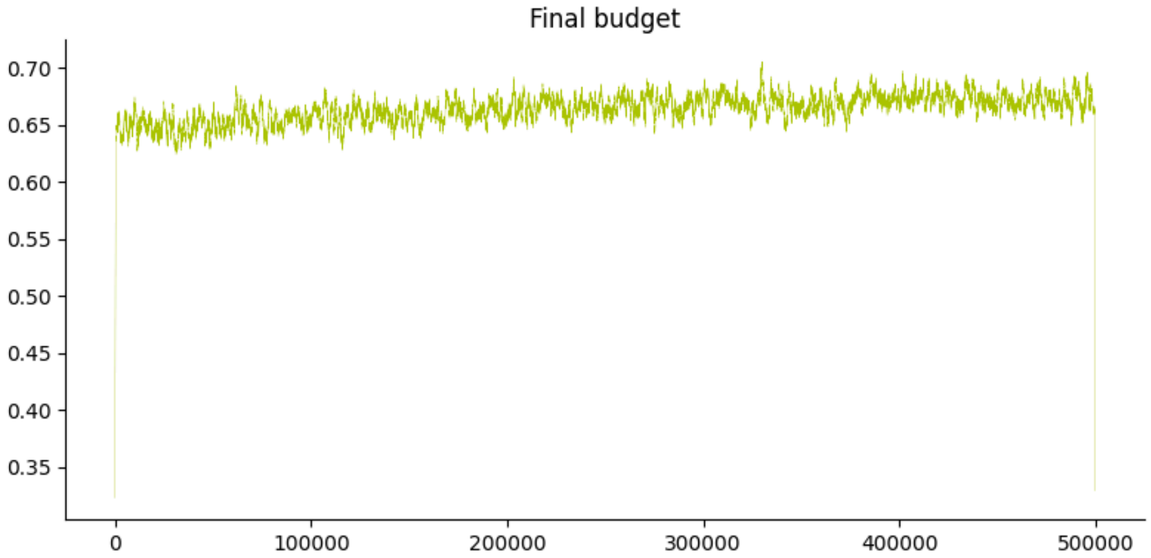


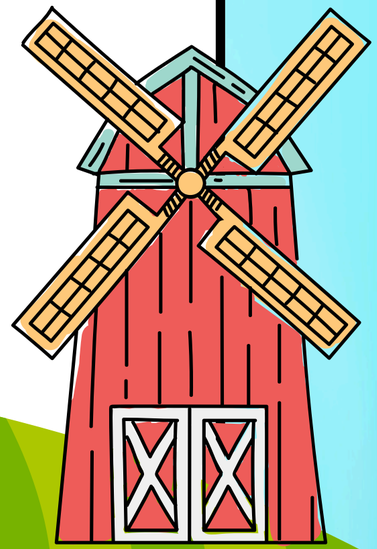


Results

The impact of the incest penalty

Incest penalty = 5 (max)

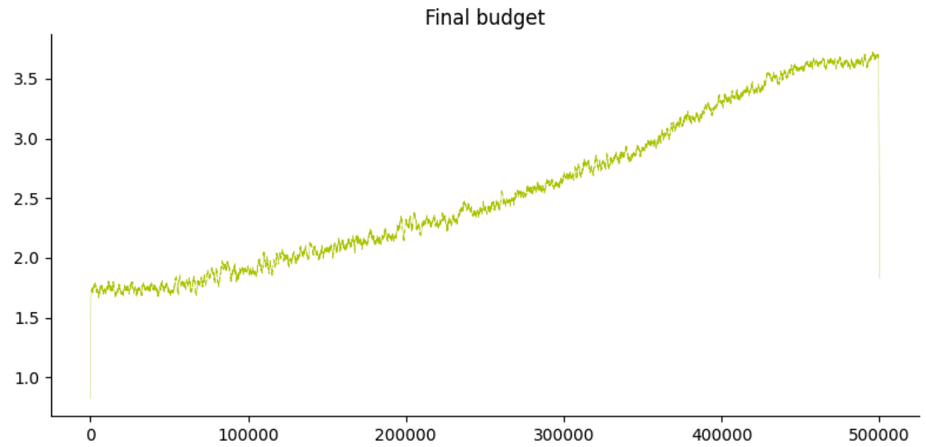




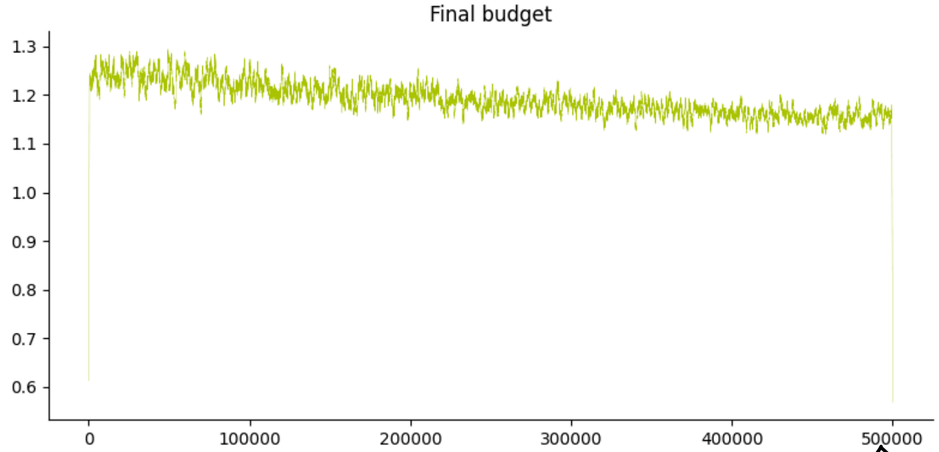
Results

The impact of the storm probability

Storm probability = 15 %



Storm probability = 60 %



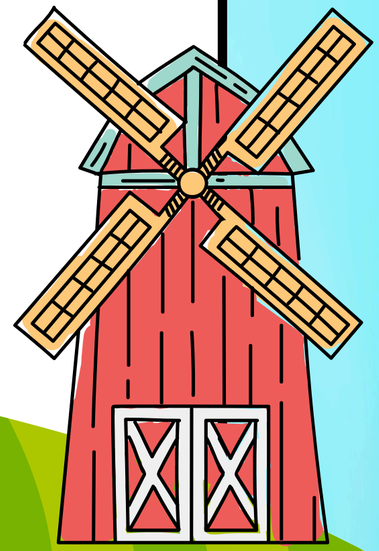
Further improvements

Environment setting:

1. Dynamic storm probability
2. More realistic sheep reproduction mechanism

Learning strategy:

1. Stabilization techniques also for VFA agents
2. Storm probability incorporated into agent knowledge



Thanks for the attention!