DRL Seminar Projekt Alain Keller

Alain Keller

June 19, 2024

1 Problem

The goal of this project was to train the ant from the Farama Gymnasium[1] to move forward in one direction. The current environment features a 8-dimensional action space with actions $a_1 \in [-1, 1]$ and a 27-dimensional observation space with $o_i \in (-\infty, \infty)$. The reward for each action is provided by the environment and cosnists of three parts. The first reward is provided every step if the ant is still alive, the second is given every n steps and rewards when the ant moved forward in the x direction and the third is the squared action vector length to penalize large movements. The rewards are weighted and summed to give a final reward.

Since the action space space is continous a classic Q-Learning approach is not suitable for this problem.

Therefore deep reinforcement learning methods were used to solve this problem. As a first approach the REINFORCE [3] was used. Later on the more modern Soft Actor Critic (SAC) [1] was used.

2 REINFORCE

The implementation of the REINFORCE Agent is in the *reinforce.py* file. To first test the agent, it was used to solve a simpler problem, the Inverted Pendulum[2]. To enforce the agent to explore new actions, an ϵ-greedy strategy was implemented to choose the new action. It can be parmaetrized with a starting ϵ, decay weight and a minimum ϵ. Every 10 iterations the current model is evaluated by running the model only with actions suggested by the trained policy. To speed up the developement, the training routine was stopped, when a perfect scrore was achieved in the evaluation. In Figures 1 and 2 the curves of the evaluation and train reward are depicted. The agent sucessfuly trained a policy to balance the pendulum. This policy can be seen in the video in *pendulum.mp4*.

Showing that the implemented agent is able to learn the goal was to use it to train a policy for the Ant environment. This step turned out to be more challenging than anticipated. With the environemnt being more complex it was expected, that more training would be needed to achieve good results. Within the given time frame the best result was a standing ant. Figures 3 and 3 show the corresponding learning curevs. The depicted training took around 18 hours. The probelm that the ant did not learn to move forward lies probably in the hyperparameters like the ϵ needed to explore. Additionally the REINFORCE actor may not be a well suited actor to train the ant due to its simplicity.

[1] `https://gymnasium.farama.org/environments/mujoco/ant/`

[2] `https://gymnasium.farama.org/environments/mujoco/inverted_pendulum/`

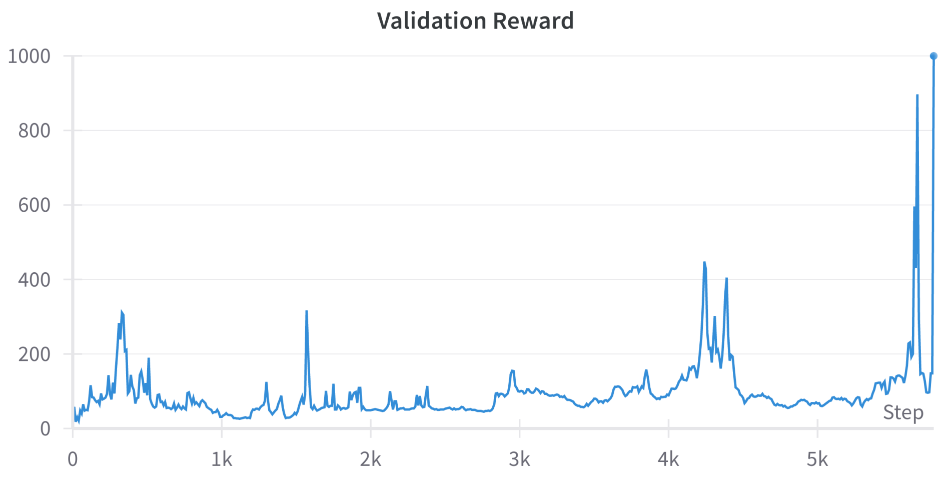

Figure 1: Evaluation run rewards vor the inverted pendulum environment

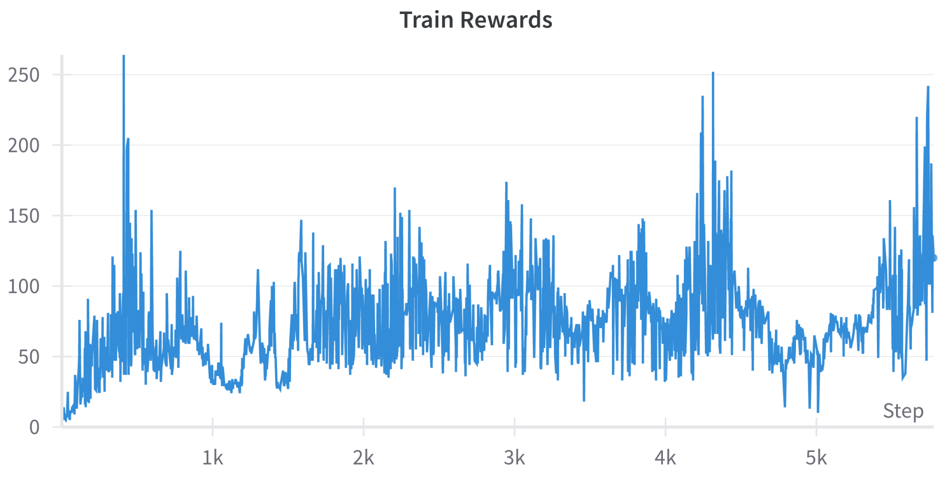

Figure 2: Training run rewards vor the inverted pendulum environment

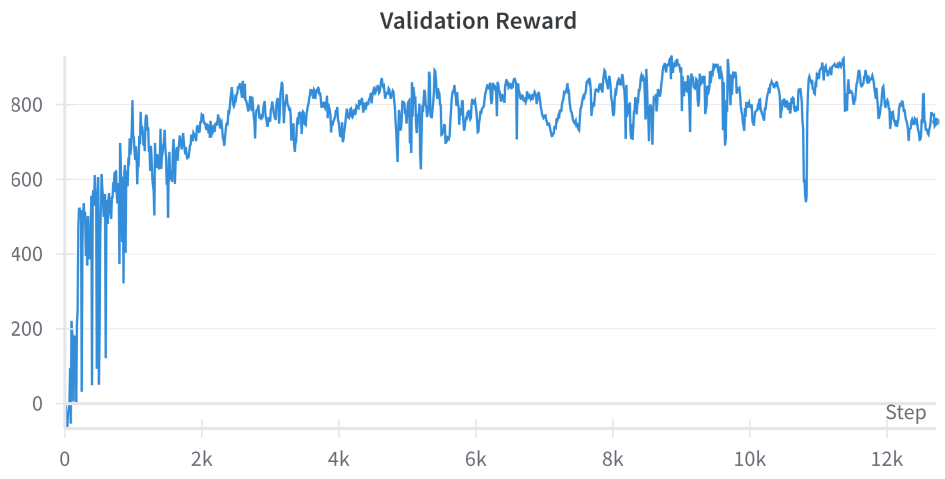

Figure 3: Evaluation run rewards vor the ant environment

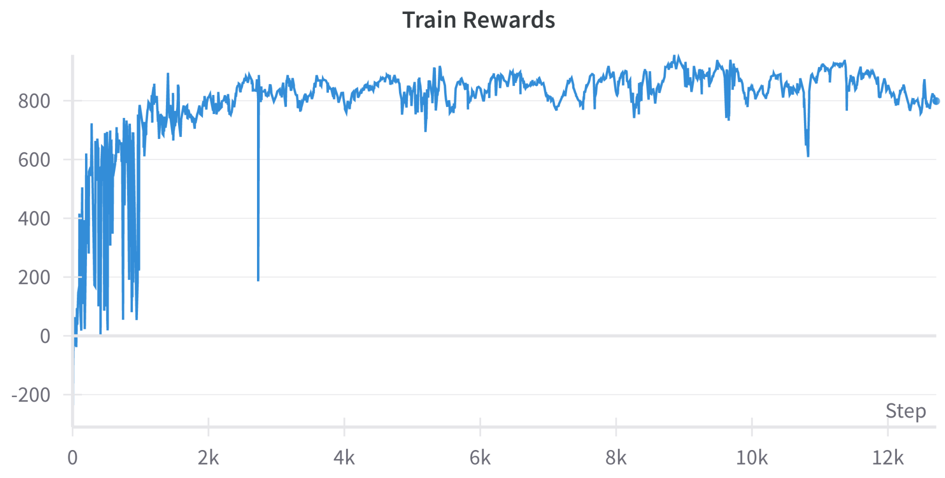

Figure 4: Training run rewards vor the ant environment.

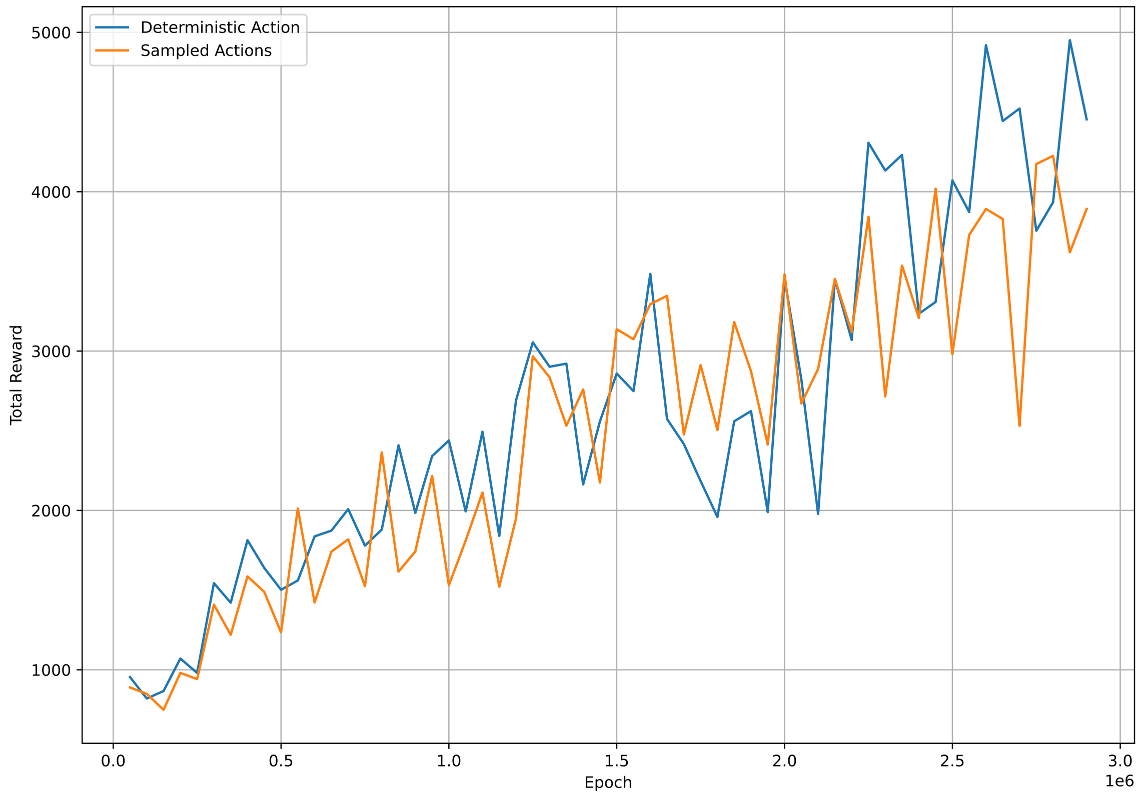

Figure 5: Training run rewards vor the ant environment.

3 SAC

As a more advanced agent, an SAC agent was implemented to train the ant. The implementation was based on the Notebook provieded in excersice 7, and [1]. A challenging part of this Agent was to set up the hyperparameters the right way. In the first expereiments a phenomenon occured, that the α parameter first went down to zero as expected, but suddendly exploded towards infinity. This was due to the numeric log probabilites of the action going towards infinity when the standard deviations in the gaussian policies are to small. The problem has been solved by changing the parameter initailization for the actor network and increasing the minimal standard deviation in the policy. With similar training lengths as with the REINFORCE agent, the SAC agent achieves much better results. Since SAC is a off policy learner, the agent can take much more training steps with the same amount of environment steps. The latter turned out to be the biggest bottleneck. However the results are not as good as the ones provided by [1].

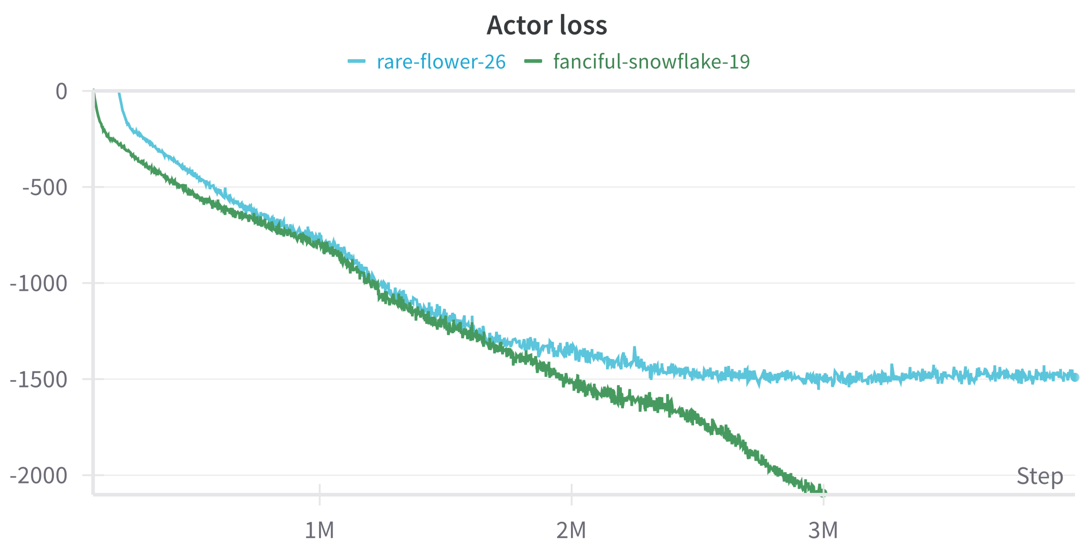

Figure 6: Actor loss, in green is the lucky run with 100'000 initial random steps and in blue the run with 200'000 inital random steps.

4 SAC

As a more advanced agent, an SAC agent was implemented to train the ant. The implementation was based on the Notebook provieded in excersice 7, and [1] with the automated entropy adjustement form [2]. A challenging part of this Agent was to set up the hyperparameters the right way. In the first expereiments a phenomenon occured, that the α parameter first went down to zero as expected, but suddendly exploded towards infinity. This was due to the numeric log probabilites of the action going towards infinity when the standard deviations in the gaussian policies are to small. The problem has been solved by changing the parameter initailization for the actor network and increasing the minimal standard deviation in the policy. With similar training lengths as with the REINFORCE agent, the SAC agent achieves much better results. Since SAC is a off policy learner, the agent can take much more training steps with the same amount of environment steps. The latter turned out to be the biggest bottleneck. However the results are not as good as the ones provided by [1]. Several trainings with different settings were made. The SAC seemed to be quite robust against minor chnages in the hyperparameters. Also the outcome was similar whether the loss is based on the learned Q-function or is an advantage consitsing of the learned Q and Value function. However one training run stood out and it was the one with the least initial random steps. Either it was luck or the initial random steps may be a hyperparameter worth finetuning. Due to the lengths of the trainings this idea could not be further investigated before submission. Figures 6, 7 and 8 show the different loss and reward curve of the 'lucky run' and a 'normal run'. The videos *blue.mp4*, *green.mp4* and *advantage.mp4* show a run of each corresponding model. The colors correspond to the colors of the figures 6, 7 and 8.

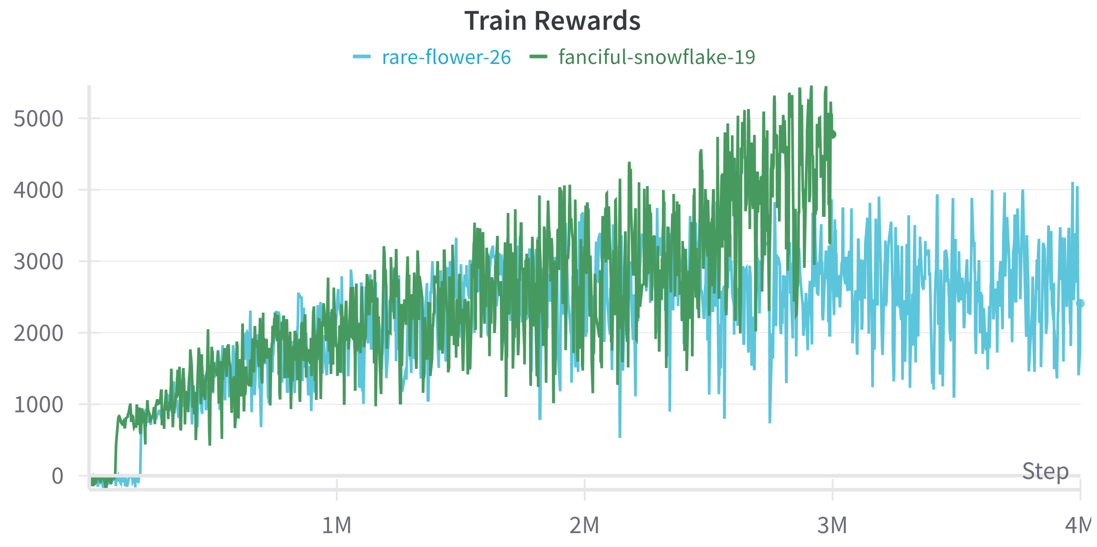

Figure 7: Training environments rewards, in green is the lucky run with 100'000 initial random steps and in blue the run with 200'000 inital random steps.

References

[1] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, 2018.

[2] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications, 2019.

[3] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992.

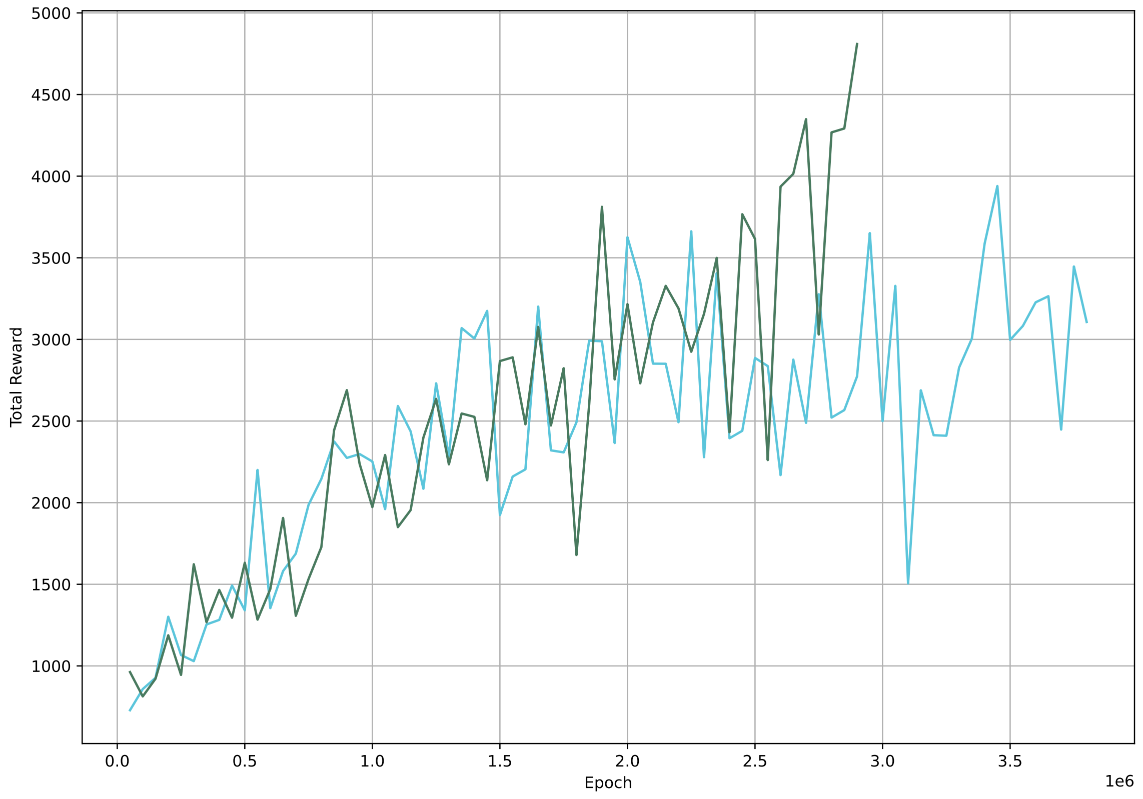

Figure 8: Validation rewards, in green is the lucky run with 100'000 initial random steps and in blue the run with 200'000 inital random steps.