Noname manuscript No.
(will be inserted by the editor)

Application of Reinforcement Learning Methods

Group 19 - Final Project Report

Yannik Frisch · Tabea Wilke ·
Maximilian Gehrke

Received: date / Accepted: date

Abstract TODO

Keywords DDPG · NAC · BallBalancer · Pendulum · DoublePendulum · FurutaPendulum

1 Introduction

We present our final results of evaluating the reinforcement learning algorithms *Deep Deterministic Policy Gradient* and *Natural Actor Critic* on the simulated quanser systems platforms *Pendulum-v1*, *BallBalancerSim-v1*, *Qube-v0*, *DoublePendulum-v0*. We also present the results of the NAC algorithm on the real double pendulum hardware system.

F. Author
first address
Tel.: +123-45-678910
Fax: +123-45-678910
E-mail: fauthor@example.com

S. Author
second address

2 Deep Deterministic Policy Gradient

The *Deep Deterministic Policy Gradient* approach [x] is an application of the *Deep Q-Learning* algorithm [x] to actor-critic methods [x] in combination with the *Deterministic Policy Gradient* [x].

2.1 Evaluation on Pendulum-v1

The OpenAI gym environment *Pendulum-v1* consists of of a singular pendulum attached on one side. The reward is depending on the angle, with a max reward of [?] per time-step for balancing the pendulum straight upright. DDPG was able to learn a good policy for this environment in less than 1000 episodes.[EDIT: PARAMETERS]

2.2 Evaluation on BallBalancerSim-v0

The Quanser Robots *BallBalancerSim-v0* environment consist of a plate whose angles can be controlled by the two dimensional input actions. The goal is to balance a ball on the plate, receiving a maximum reward of [?] per time-step for balancing it in the middle of the plate.
We started our evaluations with using the same network structures for the actor and critic as [x] did. We used 2 hidden layers with 100 and 300 hidden neurons for the actor and the critic networks and their targets. The learning rates are also set to $\alpha_{actor} = 1e-4$ and $\alpha_{critic} = 1e-3$. In figure [x] one can find our first acceptable results. The discounting is set to $\gamma = 0.99$, we did small target updates ($\tau = 1e-4$), used a mini-batch size of 64 and a total replay buffer size of 1e6. We slightly increased the noise to $\sigma_{OU} = 0.2$ and $\theta_{OU} = 0.25$ as the environments action space has an higher amplitude compared to the *Pendulum-v0*.

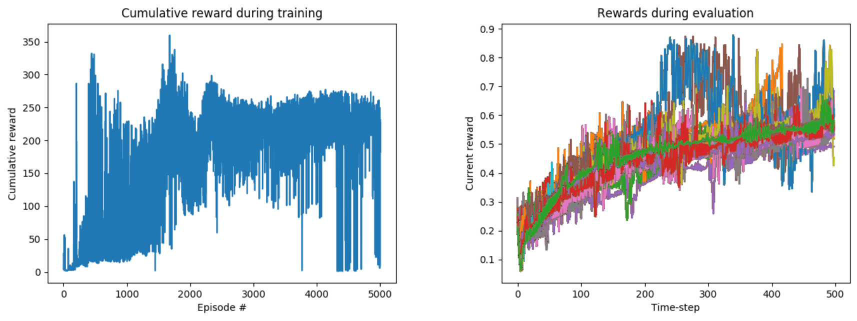

Fig. 1 The left figure shows the cumulative reward per episode during the training process. The right one displays current reward per time-step for 25 episodes of evaluation [EDIT: DO 100 EPISODES!]

The algorithm did learn to balance the ball, but was not very stable, which can also be read from the learning process plot. To further increase the stability, we increased th mini-batch size used to sample from the replay buffer, and reduced the noise again. Using weight regularization did not seem to be helpful, so we set it to zero.
Figure [x] shows the training process where discounting is set to $\gamma = 0.2$ compared to $\gamma = 0.99$. One can see discounting is crucial to solve this environment.

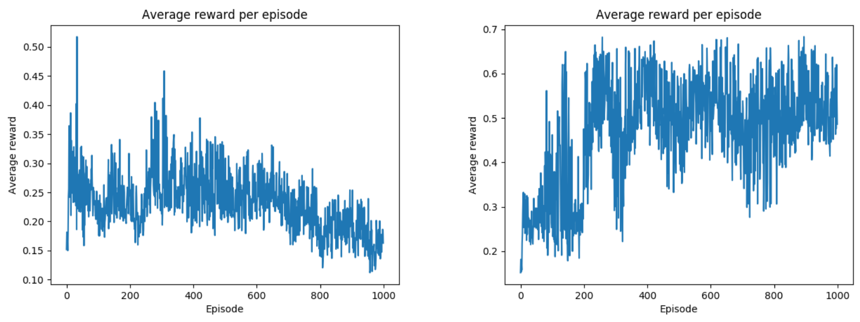

Fig. 2 The left figure shows the cumulative reward per episode during the training process with γ set to 0.2. The right one displays the process for $\gamma = 0.99$. Using discounting close to 1 was very important.

We tried to reduce the computational effort by only using a single hidden layer with 100 hidden neurons instead of two layers. The impact on the performance is shown in figure [x]. The learning suffered from instabilities, so we decided to weight the stability of using two hidden layers higher than the performance loss.

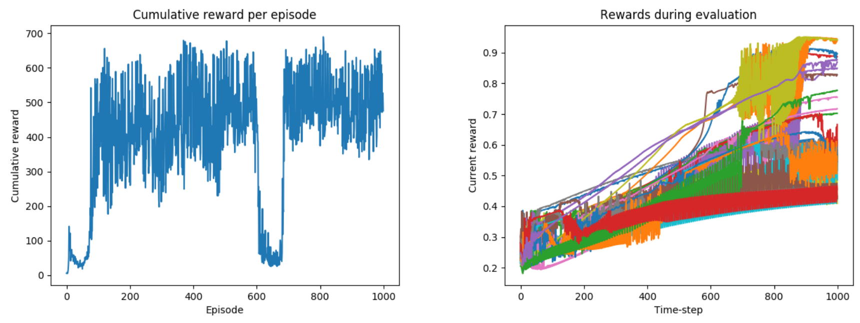

Fig. 3 The left figure shows the cumulative reward per episode during the training process using only a hidden layer and the right one again displays the performance during evaluation.

Our best results can be found in figure 4 where we set the OU action noise equal to the one used in the original paper with $\sigma_{OU} = 0.15$ and $\theta_{OU} = 0.2$. We used slightly hared updates with $\tau = 1e-3$.

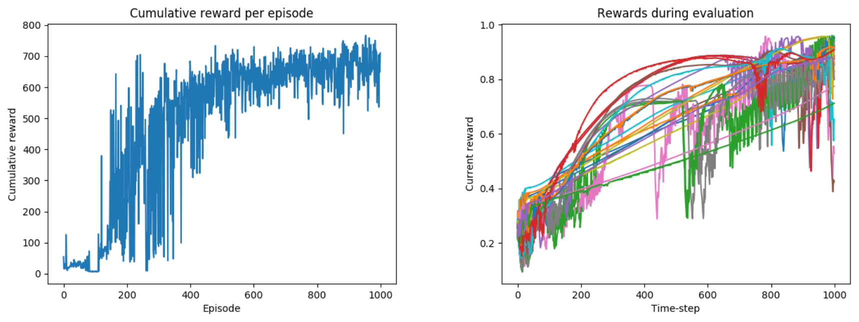

Fig. 4 The left figure shows the cumulative reward per episode during the training process. The right one displays the current reward per time-step for every evaluation episode.

2.3 Evaluation of the Pretrained Model on the Real Ball Balancer System

2.4 Evaluation on Qube-v0

The Quanser Robots *Qube-v0* environment implements the Furuta Pendulum, which consists of a motor controlling one horizontal arm. One end of the joint is attached to the motor, the other end is attached to another vertical arm, which can only be controlled indirectly by controlling the first arm. See [x] for more details.
The goal is to balance the second arm in upright position, receiving a maximum reward of 0.02 per time-step. The environment stops after 300 time-steps.

3 Natural Actor Critic

3.1 Evaluation on CartPole-v0

3.2 Evaluation on the BallBalancerSim-v0

3.3 Evaluation of the Pretrained Model on the Real Ball Balancer System

3.3.1 Learning from the Physical System

4 Discussion

References