Reinforcement Learning 2023, Master CS, Leiden University
Assignment 2 on Deep Q Learning (DQN)

Tom Stein (s3780120) [1] **Tom Stein (s3780120)** [1] **Tom Stein (s3780120)** [1]

Abstract

This assignment report focuses on Deep Q Learning (DQN) with an application to the CartPole environment. The basic concept of DQN is introduced along with the experience replay and target network improvements. A hyperparameter scan is used to empirically compare the performance of the different models and discuss the results. A high degree of instability in the training process was observed, which is, to some extent, mitigated by specific adjustments of the parameters.

1. Introduction

In this assignment, an agent has to learn how to balance a pole in the vertical position. The environment (Brockman et al., 2016) presented is a well known and defined physic problem, being a reverse pendulum. The vertical position is correspond to the unstable equilibrium point and the pole itself is attached through a join to a cart. The goal of this learning task is to keep upright the pole while moving the cart left or right. The possible action space is made up by a set of only two possible movements $0, 1$, where 0: the cart is pushed to the left; 1: the cart is pushed to the right. The state space is composed by four values: position and velocity of the cart, angle and angular velocity of the pendulum. The agent receives a reward of $+1$ for every action performed resulted in keeping the pole into a equilibrium state. The environment is reset to the initial condition every time that the value of the angle between the pole and the vertical line is bigger than 15 or when the cart leaves the range $(-2, 4; 2, 4)$.
To tackle this problem tabular methods are not sufficient anymore, since keeping in memory all the possible states is not feasible anymore. Therefore, actions cannot be performed anymore on the basis of the Q-value stored in the table, but we need a way to generalize to unseen states. This can be achieved through the application of an Artificial Neural Network. In this assignment our deep neural network taken as input a state returns the Q values for the actions.
As a baseline comparison, we will use a random policy which has a reward around 22.
The structure of the following chapters is as follows. In section 2, 3 and 4, for each of the DQN algorithm hyperparameters optimization will be analyzed, highlighting the optimal configuration. Section 2 covers the basic case of DQN algorithm while, in section 3 the replay buffer is present and in section 4 the target network is considered. After all this architecture has been presented, section 5 is dedicate the comparison and analysis of the different DQN approaches showing their pros and their limitations.

2. Methodology

2.1. DQN

2.2. Experience Replay

2.3. Target Network

3. Results

3.1. DQN with ER and TN

3.2. DQN ablation study

4. Discussion

5. Hyperparameter Scan - Bonus

In the previous sections, only individual hyperparameters were analyzed in isolation. This approach does not take into account the dependencies between the different hyperparameters such as learning rate and batch size (Smith et al., 2018). For this reason, a comprehensive hyperparameter scan was performed, in which models were trained on random combinations of the hyperparameters. In the definition of the hyperparameter space [1] no continuous value ranges like [0.8, 1.0] were intentionally used but a preselection of discrete values to be able to group and plot the results more easily. The space contains about 300 million combinations. To search this large space in a reasonable time, several parallel experiments have to be performed. For this purpose, the

*Equal contribution [1]Faculty of Science, Leiden University, Leiden, The Netherlands. Correspondence to: Tom Stein <tom.stein@tu-dortmund.de>.

[1] `wandb_sweep_config.yaml`

platform Weights and Biases (Biewald, 2020) was used to log all training runs and to orchestrate the hyperparameter scan across multiple computers. The results are shown in Appendix A, but can also be viewed interactively online[2].

The results again illustrate the instability caused by wrong parameter selection or catastrophic forgetting, since numerous runs only achieve poor results. Apart from this, there are also some positive findings, such as .....

References

Biewald, L. Experiment tracking with weights and biases, 2020. URL https://www.wandb.com/. Software available from wandb.com.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym, 2016.

Plaat, A. *Deep Reinforcement Learning*. Springer, 2022. ISBN 978-981-19-0637-4. doi: 10.1007/978-981-19-0638-1. URL https://doi.org/10.1007/978-981-19-0638-1.

Smith, S. L., Kindermans, P., Ying, C., and Le, Q. V. Don't decay the learning rate, increase the batch size. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=B1Yy1BxCZ.

A. Hyperparameter Scan Results

[2] https://api.wandb.ai/links/rl-leiden/xttn9bdo

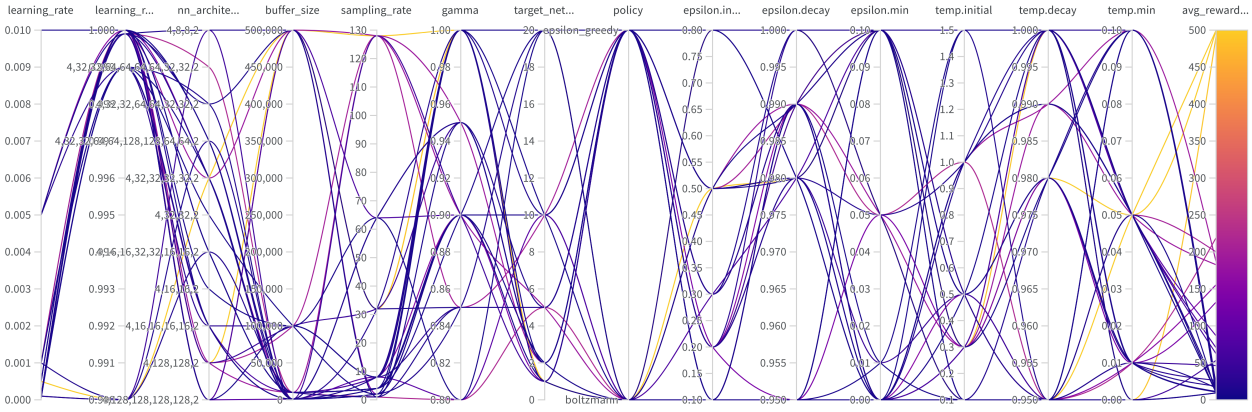

Figure 1. Parallel axis plot of the hyperparameter scan. Each line shows a single parameter configuration that was evaluated. The color indicates the performance measured by the average of the last 100 epochs (see legend on the right). Higher average reward (yellow) is better.