Machine Learning
Benchmarking neural-fitted Actor Critic with state of the art reinforcement learning algorithms[*]

H. Maathuis M. Groefsema S. Warmelink T. Oosterhuis

University of Groningen, 9747AG Groningen

Abstract

1 Introduction

With reinforcement learning an agent is placed in an environment in a state where it can execute a number of actions. Each action the agent takes in a state entails a reward or a punishment for the agent as well as bringing it to a new state. By maximizing tt's reward the agent gradually learns the value of each state (either by keeping track of this value in a lookup table, or when the environment is to complex for this to be feasible, by approximating the value function for the environment with a function approximator such as a multi-layer perceptron - MLP), allowing it to autonomously learn the optimal policy for its environment. Reinforcement learning agents can learn complex behavior this way, such as solving a maze, or obeying traffic rules in a driving simulation.
When they were first developed reinforcement learning algorithms were designed to deal with discretized state and action spaces. In real life however state spaces aren't discritized and although action spaces can be, this is not ideal as different actions may require different levels of discritization to be accurate to get the desired reward. Furthermore the required level of discritization can not be known in all cases, and having to discover it is time intensive. This potential problem can be solved by keeping the state and action space continuous in the environment representation of the agent.
Several new reinforcement learning algorithms and adaptations of existing reinforcement algorithms were developed to model continuous state and action spaces. In this paper we will benchamark one such algorithm, named Neural Fitted Actor Critic (NFAC), which was develeped in August 2016 against two established (at the time of the writing of this paper) continuous reinforcement learning algorithms, named Continuous Actor Critic Learning Automaton (CACLA) and SARSA (State Action Reward State Action) with gradient descent (referred to in this paper as GD-SARSA).
First a more extensive background will be given on reinforcement learning in general including on model free reinforcement learning amd exploration, on function approximation with MLP's and on continuous state and action spaces, then the three algorithms that are compared in this paper are explained in detail, the implementation of these algorithms and the setup of the benchmarking experiment will be discussed, after that the benchmarking results will be presented and the contribution of NFAC as a reinforecemnt learning algorithm will be evaluated.

[*]In case of an extended abstract refer to the original paper in a footnote such as "The full paper has been published in *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 13–20, 2013." Also, please keep the title and authors exactly the same as the original.

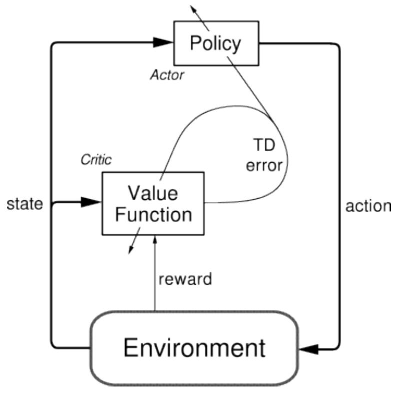

Figure 1: Actor Critic system. Reprinted from [2]

2 Background

3 Continuous Actor Critic Learning Automaton

A well established algorithm in Reinforcement Learning (RL) is Continuous Actor Critic Learning Automaton (CACLA) [1]. CACLA deals with undiscretized continuous state and action spaces. It implements an Actor Critic system in which both the Actor and Critic are operationalized using a Multi-Layer Perceptron. In an Actor-Critic system, the Actor is responsible for selecting the current action given the policy and the Critic is used in the calculation of the Temporal Difference (TD)-error which drives the learning of the Actor. The TD learning rule is characterized by equation (1), where σ_t is the TD-difference error as written in (2). The Actor-Critic system allows for a seperation between the representation of the policy and the value function. A visualization of the Actor-Critic system is shown in Figure 1. In RL problems it is important to be aware of the exploration versus exploitation trade-off; meaning that a decision has to be made between exploration which might lead to an improvement of the current policy or simply the action is selected according to the current policy. An exploration technique that is used often in CACLA is Gaussian Exploration, meaning that an action is calculated around the best action that the current policy provides. To pick a value around the best action, a sample is chosen from a gaussian distribution $N(action, standarddeviation)$. The sign of the TD-error ($\sigma_t > 0$) is used to determine whether the policy of the Actor has to be updated. To update the Actor in a connectionist way, the performed action a_t is used as the target when backpropgating the error using the state s_t as input. Since CACLA is an online RL algorithm, the data that is gathered from the interaction of the agent with the environment is used immediately to update the Critic and possibly the Actor.

$$V(S_{t+1}) = V(S_t) + \alpha_t \sigma_t \tag{1}$$

$$\sigma_t = r_t + \gamma V_t(s_{t+1}) - V_t(s_t) \tag{2}$$

4 Neural Fitted Actor Critic

Another Reinforcement Learning that uses an Actor-Critic system is Neural Fitted Actor-Critic (NFAC). Where CACLA is an offline Actor-Critic algorithm, NFAC is an online Actor-Critic algorithm. This means that a data collection D_π is built which is filled with experience tuples of the form $\{s_t, u_t, a_t, r_{t+1}, s_{t+1}\}$. s_t is the state at time t, u_t is the action the Actor MLP provides, a_t is the exploration action which can deviate from u_t, the reward at time $t + 1$ is given by r_{t+1}, and finally s_{t+1} is the resulting state after applying u_t. Every time the agent interacts in the environment, D_π will be extended with one tuple.

In CACLA, first the Critic is updated and afterwards the Actor. However since the Critic is dependent on the value of the Actor when updating in batches, the Actor is updated first. The Actor is updated towards the exploration action if the TD-error is > 0. This is identical for CACLA. NFAC deviates from

CACLA by also updating the Actor when the TD-error is ≤ 0. In that case the update of the Actor is towards the action that the Actor MLP provides. The Critic update is similar to CACLA, but in NFAC the Critic is updated for each experience in D_π. Note that the data collection needs to be emptied after every epoch since the targets used in estimating the new value function are dependent on the current value function.

5 Experimental setup

6 Conclusions and further work

7 Contributions

References

[1] H. Van Hasselt and M.A. Wiering. Reinforcement learning in continuous action spaces. In *Proceedings of the IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning, 2007*, pages 272–279. IEEE, 2007.

[2] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.