Reinforcement Learning Project
Learn Atari game *Gopher* through Deep Reinforcement Learning

Ivan Bergonzani

June 25, 2018

Abstract

In this project were tested different deep Q network architectures against the Atari 2600 game 'Gopher'. Base Deep Q network from [1] [2] together with Double Q network [3] and Dueling DQN [4] were trained on the environment provided by OpenAI Gym each for a total of 2 million frames. Despite the smaller training time with respect to the original articles, the three network were able to learn the game. They were tested over a 1000 epsiodes scoring respectively a mean reward of 150, 152, 1521.

1 Introduction

In the last few years the reinforcement learning field made great advancement and showed potential in the resolution of difficult problems. Examples of these achievement are given by the development of neural networks capable of beating Atari 2600 games as shown in [1] and following works, or by AlphaGO which is an artificial intelligence built from DeepMind that was able to beat the world champion in a complex game like GO. Moreover, reinforcement learning is used in other fields such as robotics, finance and in the medical field.

Differently from the supervised approach used for regression and classification or from the unsupervised approach exploited for clustering, in this branch of machine learning the classical situation is of an agent acting on an environment trying to maximize its performance. Formally, at each step the agent observes the current state $s \in \mathcal{S}$, performs an action from a set $\mathcal{A}$ and collect a reward $r \in \mathbb{R}$: the more the cumulative reward is over time the better is the performance. In fact, during training the agent tries to find a policy $\pi : \mathcal{S} \to \mathcal{A}$ that maximizes the value function $v : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, hence a policy that suggests the best action to take in each possible state. This policy function can be obtained through algorithms like *value iteration*, which estimates first the value function v and then build π on top of it, or like *policy iteration* that instead tries to directly compute π.

Another common algorithm is *Q-learning*, in which the training estimates a function $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ that represents how good is to take action a in a state s. The correspondent policy function will be constructed as $\pi(s) = argmax_a\ Q(s, a)$. The Q function is learned by following an iterative process where at each time t the agent perform an action a_t in the current state s_t, collects a reward r, observes the new state s_{t+1} and finally update the estimate of Q as follow:

$$Q(s_t, a_t) = (1 - \alpha) \cdot Q(s_t, a_t) + \alpha \cdot (r + \gamma \cdot max_a Q(s_{t+1}, a)) \tag{1}$$

in which α is the learning rate and γ is the discount factor in the cumulative rewards (*i.e.* how much the future rewards matter).

However, all these basic algorithms work if all the possible state-action pairs are visited infinitely often. For this reason they're used together with an ϵ-greedy strategy in order to have a good trade-off between the exploration and the exploitation phases. In the exploration phase the actions are chosen randomly while during the exploitation phase the actions are chosen using the current policy. In a ϵ-greedy strategy the action is taken randomly with a probability ϵ or it is taken following the policy with $1 - \epsilon$ probability. Usually the value of ϵ varies over time, starting with an high value to encourage exploration in the initial phases and decreasing over time to enforce exploitation.

In this work

All the experimente were executed on a laptop equipped with an Intel i7-6700HQ processor, 16GB of RAM and Nvidia GTX 960m graphics card. The networks were developed using Python and Tensorflow, while the game environment was provided by OpenAI Gym.

2 Deep Reinforcement Learning

2.1 Double Q network

2.2 Dueling Q network

3 Environment

4 Experiments

5 Conclusion

References

[1] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013.

[2] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518:529 EP –, Feb 2015.

[3] Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. *CoRR*, abs/1509.06461, 2015.

[4] Ziyu Wang, Nando de Freitas, and Marc Lanctot. Dueling network architectures for deep reinforcement learning. *CoRR*, abs/1511.06581, 2015.