Playing Atari with Deep Reinforcement Learning

Mert Bektas [1]

Abstract

With this research, the team created the first deep learning model that can play Atari games, whose input is raw pixels and does not need architecture adjustments, depending on the game.

1. Introduction

Controlling reinforcement learning (RL) agents through high-dimensional sensory inputs were always problematic. The team wondered if they could take advantage of recent advances in deep learning to extract high-level features from raw sensory data, to use in RL. On the other hand, RL adapts its data distribution as they learn new behaviors, which can be challenging for deep learning approaches that acquire a fixed underlying distribution. To overcome these data distribution problems, they used a mechanism (1) that randomly samples previous transitions so that it can smooth the training distribution. In addition, they trained the network with a variant of the Q-learning (2) algorithm with the help of stochastic gradient descent. After some tests, the network surpassed other former RL algorithms on six games while exceeding an expert human player on three.

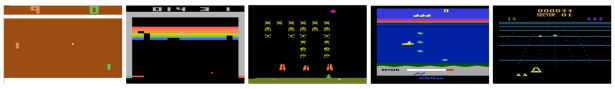

Figure 1. Screenshots from five Atari 2600 Games:(Left-to-right) Pong, Breakout, SpaceInvaders, Seaquest, Beam Rider

2. Background

At each time-step, the agent chooses an action from the set of legal game actions. Next, the agents observe nothing but an image from the Atari emulator. Also, it receives a reward, which describes the change in in-game score. The team thought to take sequences of actions and observations into account because it is impossible for the agent to understand the situation only from the current screen . For this, they applied standard RL methods for Markov decision process by using the sequences and time. The optimal action-value function adheres to the Bellman equation. To estimate the action-value function, the team applied a neural network function approximator with weights as a Q-network.

3. Related Work

Identical to Q-learning, a successful backgammon-playing program, "TD-gammon," also trained with RL and is not using a model (3). People believed that the randomness of dice rolls helped the value function to become smoother, which was seen as a coincidence. Moreover, many works in RL algorithms aimed to use linear function approximators to avoid the risk of diverging. The previous work that most closely resembles how we approached it is neural fitted Q-learning (NFQ) (4). On the contrary, our project uses RL end-to-end, getting inputs from raw images straight.

4. Deep Reinforcement Learning

The team's other goal is to efficiently process training data using stochastic gradient updates. They wanted to take TD-Gammon's approach and take it to the next level by taking advantage of hardware improvements, using modern deep neural network architectures and scalable RL algorithms. However, unlike TD-Gammon, they applied a technique known as experience replay (1), storing the agent's past experiences into a data set, which will get drawn randomly in the future. In contrast with standard online Q-learning, this technique allows for greater data efficiency, and the randomization breaks the correlation between samples, making learning more efficient.

4.1. Preprocessing and Model Architecture

The team's other goal is to efficiently process training data using stochastic gradient updates. They wanted to take TD-Gammon's approach and take it to the next level by taking advantage of hardware improvements, using modern deep neural network architectures and scalable RL algorithms. However, unlike TD-Gammon, they applied a technique

[1]Institute of Computer Science, University of Tartu, Tartu, Estonia. Correspondence to: Mert Bektas <mert.bektas@ut.ee>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

known as experience replay (1), storing the agent's past experiences into a data set, which will get drawn randomly in the future. In contrast with standard online Q-learning, this technique allows for greater data efficiency, and the randomization breaks the correlation between samples, making learning more efficient.

5. Experiments

Experiments were done with the same learning algorithm, network architecture, and hyperparameters over seven different ATARI games to see if the agent can make decisions without game-specific knowledge. Rewards are fixed to 1, -1, and 0 to scale all the games evenly. They used the RMSProp algorithm in their experiments. Ten million frames are used for training, and one million most recent frames are used as replay memory. In contrast with prior approaches to playing Atari games, they used a technique (5) where the agent takes actions and sees at intervals instead of every frame.

5.1. Training and Stability

The evaluation showed that the average predicted is smoother than the average total reward obtained by the agent. Also, they did not encounter any divergence problems during experiments. Which means their method can train stably, despite lacking theoretical convergence guarantees.

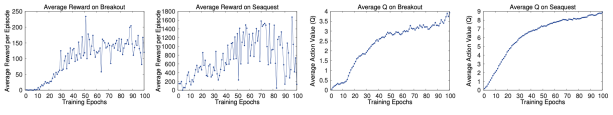

Figure 2. The two plots on the left show average reward per episode on Breakout and Seaquest respectively during training. The statistics were computed by running an epsilon-greedy policy with = 0.05 for 10000 steps. The two plots on the right show the average maximum predicted action-value of a held out set of states on Breakout and Seaquest respectively. One epoch corresponds to 50000 minibatch weight updates or roughly 30 minutes of training time.

Figure 3. The left most plot shows the predicted value function for a 30 frame segment of the game Seaquest. The three screenshots correspond to the frames labeled by A, B, and C respectively.

5.2. Visualizing the Value Function

The visualization of the learned value function proves that their method can learn how the value function adapts after a complex sequence of events.

5.3. Main Evaluation

Compared to other best-performing methods, their method has remarkably better results, although their agents get raw image inputs and have to learn independently, unlike other methods. Comparing results indicates that their method outperforms an expert human player on Breakout, Enduro, and Pong. Additionally, its performance is close to the human on Beam Rider. Besides, its performance is worse than human in other games. This is because these games expect a strategy that broadens over a lengthy period of time.

6. Conclusion

The paper presented a new deep learning model for reinforcement learning that plays Atari 2600 computer games by itself with no adjustments to the architecture or hyperparameters and gets inputs as raw frame images. In order to assist the training of deep networks for RL, they also presented an online Q-learning variation that combines stochastic minibatch updates with experience replay memory. The results are state-of-the-art in six of the seven games they tested on.

References

[1] Long-Ji Lin. Reinforcement learning for robots using neural networks. Technical report, DTIC Document, 1993

[2] Christopher JCH Watkins and Peter Dayan. Q-learning. Machine learning, 8(3-4):279–292, 1992.

[3] Gerald Tesauro. Temporal difference learning and td-gammon. Communications of the ACM, 38(3):58–68, 1995.

[4] Martin Riedmiller. Neural fitted q iteration–first experiences with a data efficient neural re- inforcement learning method. In Machine Learning: ECML 2005, pages 317–328. Springer, 2005.

[5] Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. Journal of Artificial Intelligence Research, 47:253–279, 2013.