A Comparison between Deep Q-Networks and Deep Symbolic Reinforcement Learning

Aimoré R. R. Dutra[1] and Artur S. d'Avila Garcez[1]

[1] City, University of London, London, EC1V 0HB, UK
aimorerrd@hotmail.com, a.garcez@city.ac.uk

Abstract. Deep Reinforcement Learning (DRL) has had several breakthroughs, from helicopter controlling and Atari games to the Alpha-Go success. Despite their success, DRL still lacks several important features of human intelligence, such as transfer learning, planning and interpretability. We compare two DRL approaches at learning and generalization: Deep Q-Networks and Deep Symbolic Reinforcement Learning. We implement simplified versions of these algorithms and propose two simple problems. Results indicate that although the symbolic approach is promising at generalizing and faster learning in one of the problems, it can fail systematically in the other, very similar problem. **Keywords:** Deep Reinforcement Learning, Deep Q-Networks, Neural-Symbolic Integration.

1 Introduction

The combination of classical Reinforcement Learning with Deep Neural Networks achieved human level capabilities at solving some difficult problems, especially in games with Deep Q-Networks (DQNs) [3]. There is no doubt that Deep Reinforcement Learning (DRL) has offered new perspectives for the areas of automation and AI. But why are these methods so successful? And why are they still unable to solve many problems that seem so simple for humans? Despite their success, DRL has several drawbacks. First, they need large training sets and hence learn slowly. Second, they are very task specific - a trained network that performs well on one task often performs very poorly on another, even very similar task. Third, they are difficult to extract a human-comprehensible chain of reasons for the action choices that the system makes.

Some authors have been trying to solve some of the above shortcomings by adding prior knowledge to the system, using model-based architectures and other AI concepts [2]. One claims to have designed an architecture that solves at once all these shortcomings by combining neural-network learning with aspects of symbolic AI, called Deep Symbolic Reinforcement Learning (DSRL) [1]. In this paper, in an attempt to understand better the advantages of a symbolic approach to Reinforcement Learning, we implement and compare two simplified versions of DQN and DSRL at learning a simple video game policy.

2 The Video Game

The Deep Q-Network (DQN) was reduced to a simple Q-Learning algorithm by removing its convolutional and function approximation layers. These layers do not seem to play a major role in how an agent makes its decisions. They basically reduce the dimensionality of the states. In the Deep Symbolic Reinforcement Learning (DSRL), we ignored the first low-level extraction part. In our implementation, we skip this first part by sending the location and type of each object directly to the agent. In addition, only a spatial representation is considered, since there is no complex dynamics relating to time in the game. The simplified versions of DQN and DSRL were implemented in Python 3.5.

Fig. 1 shows three initial configurations of the proposed game. The star-shaped object is the Agent, the negative sign denotes a Trap, and the positive sign is the Goal. The agent can move up, left, right and down, and it stays at the same place when it tries to move into the wall. The reward is increased by 1 and decreased by 10 whenever the Agent's position is the same as the Goal and the Trap, respectively. The game only restarts if the Agent's position is the Goal. The environment is fully-observable, sequential, static, discrete, unknown, infinite, stationary and deterministic. Two toy examples are proposed to evaluate how DQN and DSRL apply their learned knowledge in a new, similar situation, namely, training in configuration 1 and testing in 2 (c.f. Fig. 1), and training in 2 and testing in 3.

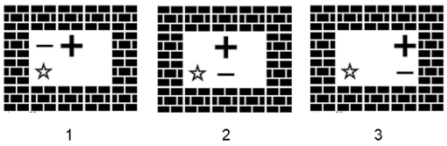

Fig. 1. Three initial game configurations

3 Results and Discussion

Fig. 2 shows that both algorithms (DQN and DSRL) learn well during the training phase, but in the test phase, while DQN has a behavior similar to random, DSRL always falls into the Trap before reaching the Goal. This shows that, while DQN could not learn from conf. 1 what to do in conf. 2; and DSRL learned something completely wrong for conf. 2 (always move to the right). It is as if any prior knowledge in DSRL had to be undefeasible, which is an unrealistic constraint. DQN, by contrast, had never seen the states in the test case during training; thus, it assumed a random policy. The reason why DSRL has very low reward is because the Goal's location did not change from training to test. Thus, our DSRL Agent

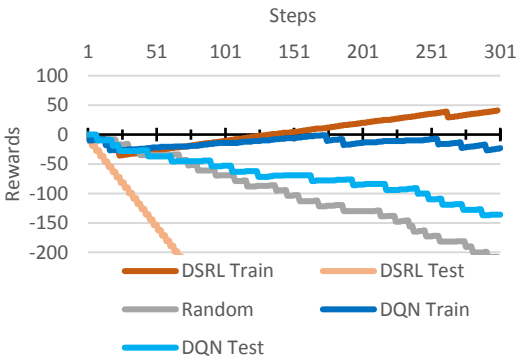

Fig. 2. Trained in conf. 1 and tested in conf. 2

assumed that the best action should remain the same (move right). The position of the Trap did not have any influence in the Agent's decision because the algorithm treats different types of objects independently. In other words, the DSRL Agent does not know what rewards to expect from a Trap in a new location.

In the second example (trained in conf. 2 and tested in 3), the situation is quite different, as Fig. 3 shows. DSRL learns how to make the right decision, and thus has good performance during testing. DQN flat lines as a result of not knowing the states in the test phase. It is interesting noting that DSRL avoided the Trap during testing because it has learned how to translate from conf. 2 to 3 (but not how to reflect from conf. 1 to 2 (c.f. Fig. 2), or to rotate a configuration, which should produce similar results as Fig. 2 for obvious reasons). Such an ability to generalize to new situations is very important, as it allows an agent to learn from similar states without having to experience them all. In the case of DSRL, generalizations bring faster learning, but seem limited to translations of configurations.

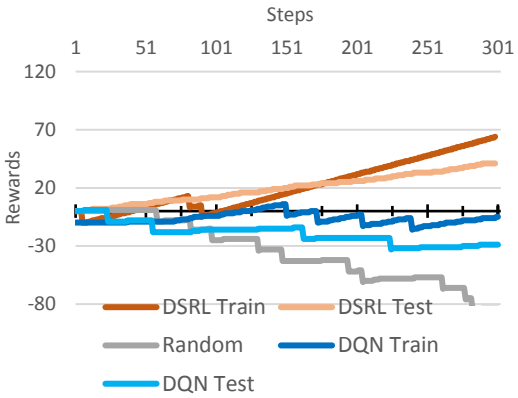

Fig. 3. Trained in conf. 2 and tested in conf. 3

4 Conclusion

We have compared two model-free RL approaches, DRL and DSRL, on their generalization capacity using two toy examples. Both have limitations at learning "the rules of the game" for succeeding in different configurations. One key finding is that transforming pixels into symbols can become a channel not only for reducing the state-space, but to enable rules between objects to be created. These rules offer a way of generalizing states, and could guide an agent during exploration. Assisted by high level rules, an agent should learn faster by exploring its environment more efficiently. Thus, as future work, we shall consider the combination of model-free and model-based approaches with symbolic rules being used for faster and hopefully more effective learning.

References

[1] Marta Garnelo, Kai Arulkumaran, and Murray Shanahan. Towards deep symbolic reinforcement learning. *arXiv preprint arXiv:1609.05518*, 2016.

[2] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[3] Volodymyr Mnih, Koray Kavukcuoglu, David Silver et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.