ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0.
Available from http://www.i6doc.com/en/.

On the Speedup of Deep Reinforcement Learning Deep Q-Networks (RL-DQNs)

Anas M. Albaghajati[1] and Lahouari Ghouti[1] *
1- King Fahd University of Petroleum and Minerals - Department of Information and Computer Science. KFUPM Box 1128. Dhahran 31261 - Saudi Arabia

Abstract. Deep reinforcement learning (DRL) merges reinforcement (RL) and deep learning (DL). In DRL-based agents rely on high-dimensional imagery inputs to make accurate decisions. Such excessively high-dimensional inputs and sophisticated algorithms require very powerful computing resources and longer training times. To alleviate the need for powerful resources and reduce the training times, this paper proposes novel solutions to mitigate the curse-of-dimensionality without compromising the DRL agent performance. Using these solutions, the deep Q-network model (DQN) and its improved versions require less training times while achieving better performance.

1 Introduction

Reinforcement learning (RL) allows artificial intelligence (AI) agents to learn certain tasks by trial-and-error to achieve the maximum possible reward. These agents keep adapting their actions to develop the "*best*" policy. On the other hand, state-of-the-art deep learning (DL) algorithms, combined with RL concepts, have shown great capability. This combination, commonly known as deep reinforcement learning (DRL), has been publicized by Google DeepMind thanks to their deep Q-network (DQN). DQN astutely combines convolutional neural networks (CNNs) with RL models [12]. Several improvements and refinements of the DQN model have been proposed to boost its performance [7] [15]. However, most of these improvements focus on enhancing the agent intelligence. The latter is tightly connected to the algorithmic sophistication and CNN model architecture. However, this sophistication comes at the cost of an increased computational complexity. In this case, agent training can take from several hours to several days or even over a month. Such limitation does not favor researchers and developers with low and moderate computing resources. To alleviate this limitation, it is necessary to develop strategies to reduce the training times by orders of magnitude. At the same time, this reduction should maintain comparable intelligence capabilities compared to sophisticated agents. It is worth mentioning that Van Hasselt et al. [7] improved the DQN model without causing an increase in its computational complexity. To speed up the training time of the DRL agent, Mnih et al. [13] used multithreading techniques to increase the algorithm efficiency.

* This work is supported by King Fahd University of Petroleum and Minerals.

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0. Available from http://www.i6doc.com/en/.

2 Review of Existing RL-Based Algorithms

TD-Gammon represents one of the earliest combinations of RL concepts and artificial neural networks (ANNs) [6]. TDGammon was trained using random initialization and a temporal-difference RL (TD-RL) routine through a self-play strategy. Despite its performance, yet it depended on human intervention for the *hand-crafted rules*. In 2013, Google DeepMind proposed a groundbreaking DRL-enabled model, a non-linear value function approximator, to autonomously learn control policies [12]. At that time, existing RL models relied on hand-crafted features. DeepMind resorted to a strategy called *experience reply mechanism* to account for data correlation and non-stationary distributions. In [11], Schaul et al. proposed the use of a prioritized experience reply buffer to rank experiences based on the rewards achieved. The double Q-network algorithm, built on top of Q-learning, overestimates action values although showing better performance when learning Atari games. According to Van Hasselt et al. [7], the DQN algorithm can surpass human experts when the action value estimates are close to the true ones or underestimated. Another improvement to the DQN model is commonly known as the double DQN (DDQN) is adapted to work with non-linear function approximators. These approximators are deep CNNs layers. Experimental results have indicated that the DDQN-based agent outperformed that based on the DQN model when playing Atari games. The dueling DQN, another improvement of the DQN model, is suggested by Wang et al. [15]. The improvement consisted of a new CNN architecture that leverages two streams. Since the dueling DQN consists of layers with sizes similar to those in the DQN model, it can take advantage of any improvements applied to the original DQN such as those suggested in the DDQN model. Hausknecht and Stone [8] replaced the first post-convolutional fully-connected layer in the DQN model with a recurrent long short-term memory (LSTM) layer. This solution added a recurrence property to the DQN model which resulted in the deep recurrent Q-network (DRQN). All the previous DRL solutions have been trained assuming a two-dimensional (2D) environment where the states are *fully-observable*. Lample and Chaplot proposed a new learning technique to tackle a more challenging three-dimensional (3D) environments in first-person shooter games [5]. Alvernaz and Togelius reduced the high-dimensional imagery inputs into lower-dimensional representations using auto-encoders [10]. The performance of their model is assessed using the VizDoom environment where it showed promising results. Guo et al. noticed that planning-based agents can achieve higher results than the best model-free ones [14]. Model-free agents are slower and not suitable for real-time play. They are also unrealistic as they rely on information not available to human players. *AlphaGo*, a recent achievement by Google DeepMind, is able to master the *Go* game. It is worth mentioning that this game has been considered the most challenging classical game for AI-based agents [1]. In Go game, the number of states exceeds 10^{170} [1]. The success recipe of *AlphaGo* consisted of combining several AI and ML algorithms and making them work together. In [3], Silver et al. introduced a new approach for merging DL and DRL with *Monte Carlo Tree Search* (MCTS). This merger allowed the agent to learn the best policies in playing the *Go* game. *AlphaGo* achieved a 99.8% winning rate against these agents and defeated the top two human players in the world. To go beyond the rewards provided by the environment, Pathak, et al. [2] proposed an agent capable of developing self-motivation and curiosity to

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0.
Available from http://www.i6doc.com/en/.

explore new states. The self-motivation and curiosity components are assumed by the intrinsic curiosity module (ICM).

3 Proposed Enhanced Q-Network

3.1 Observability Issue

Games in the arcade learning environment (*ALE*) and *ViZDoom* are considered partially-observable as the DRL-agent is fed each time with a single game frame. One game frame can be very ambiguous to the DRL-agent as illustrated in Fig. 1. Fig. 1 indicates that the agent can only speculate about the ball direction, movement and speed in a typical Pong game.

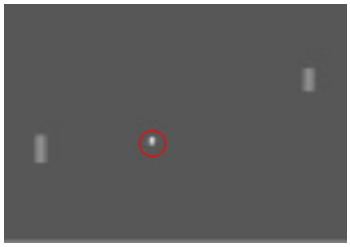

Fig. 1: Illustration of state ambiguity in game environments (ball circled in red).

3.2 Proposed CNN Simplification Technique: Step 1

This paper proposes an effective simplification procedure for the CNN layer of the DQN model. Our solution is as affective as the framing stacking one, yet it leads to a simpler CNN architecture. More specifically, instead of feeding the CNN layer with a sequence of frames, a merged version of the same sequence is used. The merged version yields the CNN layer with information similar to that captured by the frame sequence. However, simple frame merging can lead to degraded overall performance compared to even a single frame as shown in the left of Fig. 2. Therefore, the DRL-based agent will suffer from more ambiguity. To eliminate this ambiguity, we suggest to reduce the pixel intensity of the frames as they get older in the agent memory buffer. Intensity reduction will allow the agent to properly infer the ball speed and direction in the game.

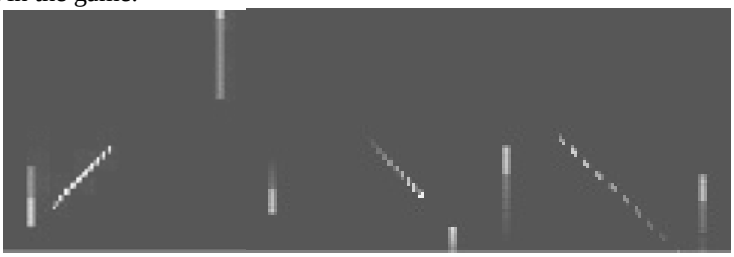

Fig. 2: Effect of merging four frames in Pong game (left). Inference of ball speed (center) and direction (right).

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0.
Available from http://www.i6doc.com/en/.

3.3 Proposed CNN Simplification Technique: Step 2

The frame background color causes the information to fade as the data moves deeper in the CNN layer. This will reduce the effectiveness of the frame merging technique. To let the CNN layer focus on the game dynamics, we suggest to eliminate the background information from any computations taking place in the CNN layer. This will ensure that only relevant information is processed. In Pong game for example the only thing to be left are the two paddles and the ball as shown in Fig. 3.

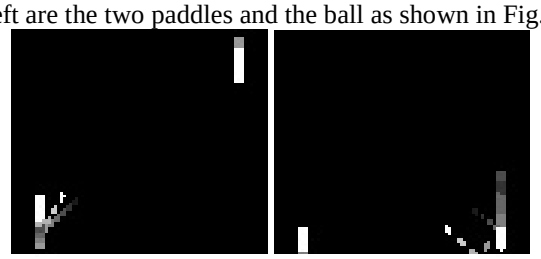

Fig. 3: Merging technique with background removal in Pong game.

4 Experimental Results

4.1 Experiment Design and Setup

All the experiments were conducted on a Linux machine running Ubuntu OS v16.04LTS with a 3.2 GHz Intel i7-6900K Octa-core, 64 GB RAM, Nvidia Geforce GTX 1080 (8 GB). All computer simulations are conducted under *Gym* 0.9.4 [4] and *ViZDoom* [9] environments.

4.2 Performance Results

The *ALE* environment Pong game is used to train the DRL-based agents. First, we replaced the frame stack with a single frame in the CNN layer. Fig. 4 summarizes the performance and training times of the resulting agent. While using a single frame causes a considerable performance degradation, the training time is reduced by 16%. This reduction hints that improving the quality and richness of a single frame, fed to the CNN layer, would boost the performance of the game agent as suggested in our paper.

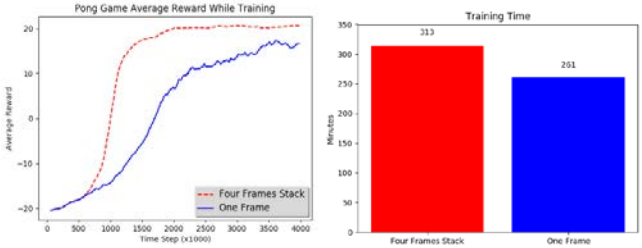

Fig. 4: Game average rewards (left) and training times (right) using frame stacks and single frames.

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0.
Available from http://www.i6doc.com/en/.

Next, ten frames, merged using our proposed solution, are fed to the CNN layer. The performance and training times of the resulting agent are shown in Fig. 5. As expected, our solution exhibited an improved performance over the one-frame scheme while achieving less training time by 10%.

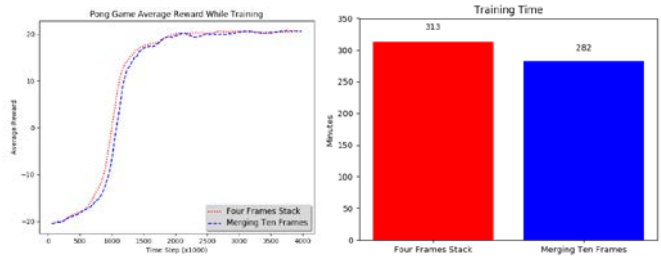

Fig. 5: Game average rewards (left) and training times (right) using frame stacks and frame merging.

To improve the training performance, our improved merging solution removes the background information from the visual information fed to the CNN layer. Fig. 6 shows the performance of the simplification solution combined with the background removal. Our solution did not only improve the model performance but it has also reduced the training time by 18%.

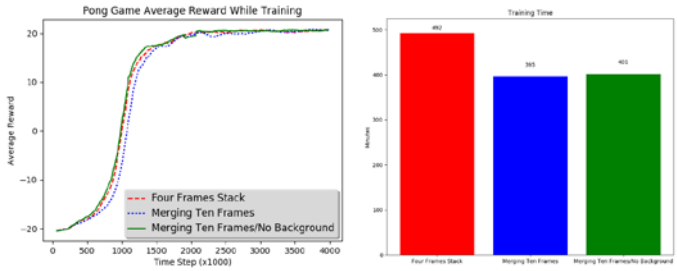

Fig. 6: Game average rewards (left) and training times (right)using frame merging with background removal.

Next, our solutions are applied to two improved versions of the DQN model (DDQN and dueling DQN). These two modified versions have achieved a reduction in training time of 23%.

5 Conclusions

In this paper, two efficient modifications of the CNN layer in DRL-based agents have been proposed. The environment features are captured by merging several consecutive raw frames of the game board and fed to the CNN layer. These frames are processed though background removal and pixel fading to extract valuable temporal information about the game states. Thanks to these solutions, the deep Q-network model (DQN) and its improved versions require less training times while achieving better performance when playing games such as *Pong* and *VizDoom*.

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0. Available from http://www.i6doc.com/en/.

Complex DQN versions such as the double DQN and dueling DQN require 20% less training times while outperforming their non-enhanced counterparts.

Acknowledgement

The authors would like to acknowledge the support provided by King Fahd University of Petroleum & Minerals (KFUPM).

References

[1] A. Nandy and M. Biswas, "Reinforcement learning with Keras, TensorFlow, and ChainerRL," in Reinforcement Learning. Springer, 2018, pp. 129–153.

[2] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," arXiv preprint arXiv:1705.05363, 2017.

[3] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot et al., "Mastering the game of go with deep neural networks and tree search," Nature, vol. 529, no. 7587, pp. 484–489, 2016.

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

[5] G. Lample and D. S. Chaplot, "Playing fps games with deep reinforcement learning." in AAAI, 2017, pp. 2140–2146.

[6] G. Tesauro, "Temporal difference learning and td-gammon," Communications of the ACM, vol. 38, no. 3, pp. 58–68, 1995.

[7] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning." in AAAI, 2016, pp. 2094–2100.

[8] M. Hausknecht and P. Stone, "Deep recurrent q-learning for partially observable mdps," CoRR, abs/1507.06527, 2015.

[9] M. Kempka, M. Wydmuch, G. Runc, J. Toczek, and W. Jaśkowski, "Vizdoom: A doom-based ai research platform for visual reinforcement learning," arXiv preprint arXiv:1605.02097, 2016.

[10] S. Alvernaz and J. Togelius, "Autoencoder-augmented neuroevolution for visual doom playing," arXiv preprint arXiv:1707.03902, 2017.

[11] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," arXiv preprint arXiv:1511.05952, 2015.

[12] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski et al., "Human-level control through deep reinforcement learning," Nature, vol. 518, no. 7540, pp.529–533, 2015.

[13] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in International Conference on Machine Learning, 2016, pp. 1928–1937.

[14] X. Guo, S. Singh, H. Lee, R. L. Lewis, and X. Wang, "Deep learning for real-time atari game play using offline monte-carlo tree search planning," in Advances in neural information processing systems, 2014, pp. 3338–3346.

[15] Z. Wang, T. Schaul, M. Hessel, H. Van Hasselt, M. Lanctot, and N. De Freitas, "Dueling network architectures for deep reinforcement learning," arXiv preprint arXiv:1511.06581, 2015.