Optimal Use of Experience in First Person Shooter Environments

Matthew Aitchison
College of Engineering and Computer Science
Australian National University
Canberra, Australia
Email: Matthew.Aitchison@anu.edu.au

Abstract – Although reinforcement learning has made great strides recently, a continuing limitation is that it requires an extremely high number of interactions with the environment. In this paper, we explore the effectiveness of reusing experience from the experience replay buffer in the Deep Q-Learning algorithm. We test the effectiveness of applying learning update steps multiple times per environmental step in the VizDoom environment and show first, this requires a change in the learning rate, and second that it does not improve the performance of the agent. Furthermore, we show that updating less frequently is effective up to a ratio of 4:1, after which performance degrades significantly. These results quantitatively confirm the widespread practice of performing learning updates every 4th environmental step.

Index Terms – Deep Learning, Game AI, Reinforcement Learning, Experience Replay

I. Introduction

Reinforcement learning (RL) in games has gained a lot of attention recently due to some high-profile successes such as super-human performance in the Atari games [1], StarCraft [2], and the game of Go [3]. However, these algorithms require an extraordinary amount of experience to train the agents. AlphaZero, for example, played almost 5-million games to reach peak performance. This level of sample inefficiency limits the application of these algorithms to real-world problems. In situations where generating experience is costly, for example when robotic interaction with the world is required, episodes typically measure in thousands rather than millions [4]. Interest has been building for sample efficient algorithms with the introduction of the MineRL environment [5], as this is of critical importance to the application of RL to real-world robotic problems.

The Deep Q-Learning Network (DQN) algorithm [1] first demonstrated that Deep Neural Networks (DNNs) could be successfully applied to complex RL tasks. To do this, thr algorithm implemented an Experience Replay (ER) [6] system that allowed the agent to sample randomly from previously experienced transitions. Although the primary purpose was to decorrelate the samples, a useful side effect is the ability to learn from experience multiple times. The original paper performed one learning step on 32 samples for every environmental step, meaning that each transition would be seen an average of 32 times. Later papers instead apply updates every 4th environment step giving transitions an expected sample rate of 8 times [7] [8].

978-1-7281-1884-0/19/$31.00

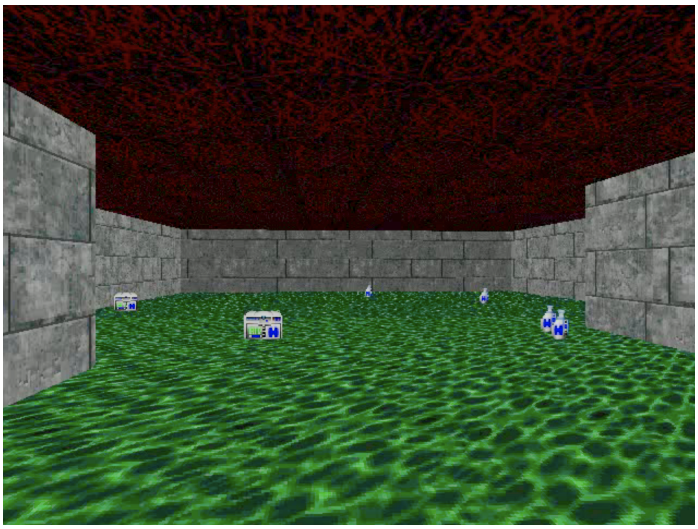

Figure 1. ViZDoom 'Health Gathering Supreme' scenario showing health kits and poison bottles.

To our knowledge, this change has never been quantitatively assessed in a complex 3D environment. In this paper, we investigate the impact of adjusting the learning update ratio for DQN under a First Person Shooter 3D environment. If increased sampling could make up for environmental steps training schemes may be able to apply additional model updates in lieu of environmental steps where the latter has a significantly higher cost. Furthermore, in situations where the cost of generating experience is relatively low, a lower frequency may improve training times with little impact on the performance of the agent.

The 'Health Gathering Supreme' scenario in the ViZDoom environment was used to evaluate agents performance under different experience replay sampling ratios [9]. The scenario tests navigating and fine-grained control in a challenging, rich, 3D environment as shown in figure 1. Furthermore, the graphics of ViZDoom more closely represent a real-world setting than other environments such as the Atari Learning Environment (ALE) [10].

II. Related Work

Various sample efficient algorithms have been proposed to address the shortcomings of the DQN algorithm. These algorithms, however, would also benefit from knowing the optimal ratio between taking environmental steps and learning steps.

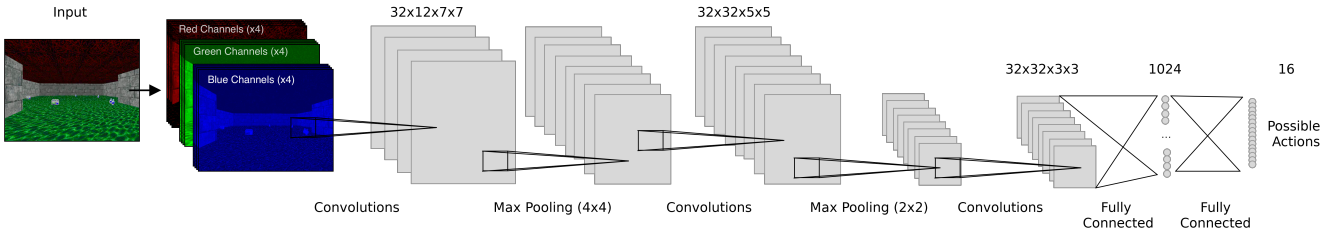

Figure 2. Architecture of the model used in these experiments.

Table I
HYPERPARAMETER SETTINGS

Hyperparameter	Value
Minibatch size	32
Replay memory size	10,000
Target network update frequency	1,000
Discount factor	1.0
Initial exploration rate	1
Final exploration rate	0.1
Frame skip	10

A. A3C / A2C

The Asynchronous Advantage Actor-Critic (A3C) algorithm and its synchronous version A2C were designed to improve the computational efficiency of the Actor-Critic algorithm by allowing agents to run in parallel [11]. This greatly improves training time but has poor sample efficiency due to the lack of an experience replay.

B. Prioritized Experience Replay

Prioritized experience replay reuses experiences from the past according to their importance [12]. This importance is estimated from the temporal difference (TD) error, where larger errors are believed to be more important. Importance sampling greatly increases the efficiency of an experience replay buffer, however the question of how many times to sample the replay buffer remains.

C. ACER

The Actor-Critic with Experience Replay (ACER) algorithm extends A2C to use an experience replay and shows that mixing off-policy experience replay updates with on-policy updates provides much better sample efficiency [13]. This approach differs from ours in that is mixes off-policy updates with on-policy, while ours uses only off-policy updates.

III. METHOD

In this section, we outline the experimental method for evaluating the optimal learning step ratio in the ViZDoom 'Health Gathering Supreme' environment. Where the learning step ratio is the number of times to apply a learning update per environmental step. Ratios less than 1 indicate multiple environmental steps per learning update.

A. ViZDoom Environment

The environment selected was 'Health Gathering Supreme' which provides a substantial challenge to both AI agents and human players alike [9]. The scenario involves navigating a maze while collecting health kits and avoiding poison bottles (see figure 1). At intervals, the agent's health is decreased making it imperative to obtain health kits continuously. The scenario settings were left unchanged.[1]

B. Reward Shaping

We use change in health as the reward function as per [14], as well as adding an auxiliary reward of +100 for each health pack and -100 for each poison bottle. As health is capped at 100, this change slightly reduces the reward for obtaining health packs at high health levels. The final score for the agent is calculated as the average health over the episode, with any time spent dead counted as 0. This better distinguishes between an agent who maintains high health over the entire episode from one who only just survives. Comparison is also made with Kempka et al's original agent according to the default scoring system. Using our scoring system an idle agent receives an average score of 11.6, whereas a theoretically perfect score would be close to 100.0.

C. Network Architecture

The network architecture used is nearly identical to the model used in [9] and is described in figure 2.

D. Hyperparameters

Hyperparameters were left largely unchanged from the ViZDoom paper [9], with the following changes: A minibatch size of 32 was used to better match the DQN paper. Also, a target network updating scheme was implemented as per [1], as this was found to improve training stability. A full list of hyperparameters can be found in table I. For each learning step ratio, α, a learning rate search was performed by selecting learning rates from

$$\{5 \cdot 10^{-5} \cdot \alpha \cdot 2^k \mid k \in [-2, -1, 0, 1, 2]\} \quad (1)$$

and choosing the learning rate with the highest final score.

Even though the VizDoom paper used 1,000,000 environment steps to train their model we found that by including target updates, most of our models converged much faster than this

[1]The VizDoom paper uses a backwards action which is not part of the default configuration for this environment, so one was added. Also, the screenshots from the VizDoom paper appear to use a different texture pack as the poison bottles have a different colour. The texture of the bottle in ViZDoom V1.1.7 is more difficult to distinguish from health than the reference screenshot in their paper

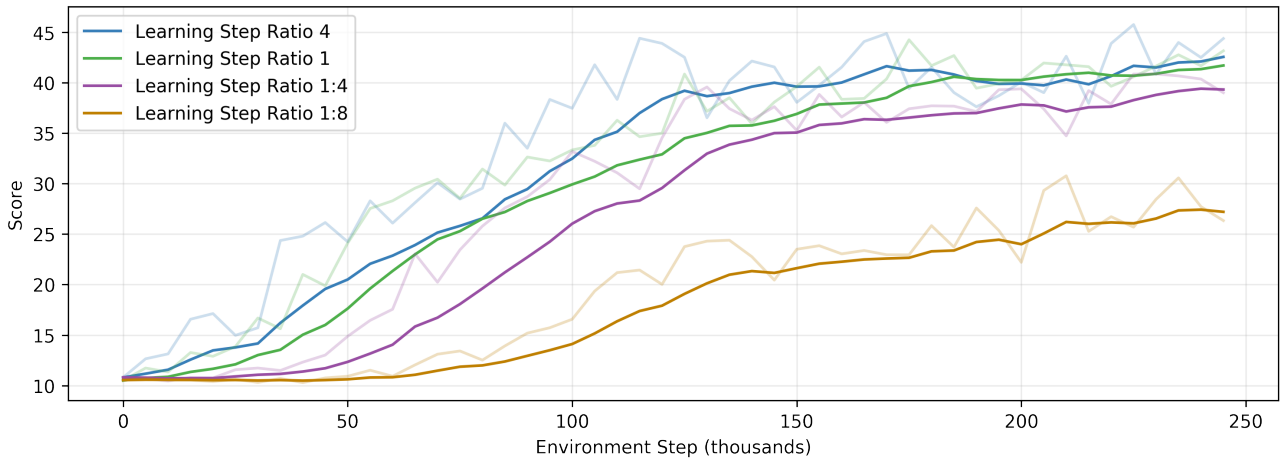

Figure 3. Agents test scores during training taken from the optimal learning rate, and averaged over the 5 runs. The every 4th step agent takes longer to start, but ends up only slightly underperforming the other agents even with significantly fewer learning updates. Raw scores are shown faded, with a smoothed with an exponential moving average ($\epsilon = 0.8$) overlaid.

and did not improve much after the 150,000 mark, so trained for 250,000 environment steps. The shorter environment steps helped with training times, and better represents a scenario in which generating environment steps is expensive.

E. Evaluation

Agents were evaluated by running the agent through the scenario 25 times every 5,000 environment steps, each time starting from a random position and orientation in the maze. The scores over these episodes are then averaged. Due to fluctuations over time the results are then smoothed using an exponential moving average (EMA) with $\epsilon = 0.8$, and the top 10% results averaged to give a final score. Results for each experiment were repeated 5 times with the average of these scores being presented.

IV. Experiments

A. Experimental Setup

Experiments were performed with ViZDoom 1.1.7, on an Nvidia P100 GPU using PyTorch. We used the RMSProp optimizer with various learning rates and trained for 250,000 environmental steps. Target model updates were performed every 1,000 learning steps. Training times ranged from 1 to 12 hours depending on the learning step ratio used.

B. Experimental Results

Table II
Test results for agents with various learning step ratios.

Learning Ratio	Learning Rate	Score	Reward
4:1	1.25×10^{-5}	42	1142
2:1	2.5×10^{-5}	41	1132
1:1	$\mathbf{5 \times 10^{-5}}$	**42**	**1163**
1:2	2×10^{-4}	41	1129
1:4	4×10^{-4}	38	1015
1:8	2×10^{-4}	31	903
1:16	4×10^{-4}	24	699
1:32	2×10^{-4}	18	538

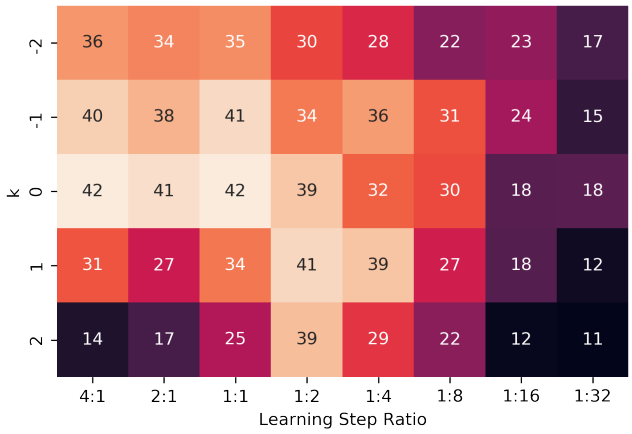

Figure 4. Heat map showing average final score for of the 5 runs over each learning rate modifier k from eq. 1 and learning step ratio. A learning step ratio of 4:1 indicates 4 learning updates per environmental step, whereas a ratio of 1:4 indicates learning updates every 4th step.

The agent's performance for each learning step ratio, is given in table II. Learning rate is the optimal learning rate tested, score the average final score over the 5 runs, and reward the average final reward over the 5 runs as per the ViZDoom paper. The agents performed similarly to those tested by Kempka et al, with the best run of the learning update every 4th environment step agent scoring 1,374 compared to approximately 1,300 in the original paper. The average results, however, are brought down by some runs failing to converge well.

Figure 3 shows the average test performance during training for learning step ratios 4:1, 1:1, 1:4 and 1:8. Applying learning updates every 4th step does initially slow down training, however, this effect is minimal after 200,000 environment steps. Applying learning updates every 8th step, on the other hand, significantly degrades performance.

Figure 4 records the average score over the 5 runs for each learning rate modifier k and learning step ratio. These results show that an increased learning step rate, by its self, is not sufficient to improve the performance of the model. Indeed,

simply applying learning updates additional times leads to a rapid decrease in performance. However, if the learning rate is similarly decreased by the equivalent amount, the agent's performance remains comparable. Notably, the learning step ratio can also be decreased to every 4th environment step with little impact on performance, again, so long as the learning rate is adjusted accordingly. The reason for the agent's sensitivity to the learning rate is not clear. One possible explanation could be that a certain amount of progress is needed per environment step, and it does not matter if this progress performed with multiple small steps or one large step.

V. Conclusions and Future Work

These results show that increased sampling of experience from the experience replay buffer does not improve an agent's performance, nor does it increase the speed at which an agent learns. On the other hand, decreasing the learning step ratio to a rate of every 4th environmental step has only a small negative effect on performance. Reductions beyond this point, however, severely degrade the agent's performance. When adjusting the learning step ratio it is important to also adjust the learning rate by an equal amount to maintain optimal performance. These results quantitatively support for the 'rule of thumb' of applying updates every 4th environmental step, with only a minimal impact on performance. These results suggest further research into the effect of learning update ratios on other algorithms that make use of Experience Replay, such as the ACER algorithm.

Acknowledgement

This research was supported by an Australian Government Research Training Program (RTP) Scholarship.

References

[1] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning", *Nature*, vol. 518, 2015. DOI: 10.1038/nature14236. [Online]. Available: https://www.nature.com/articles/nature14236.pdf.

[2] S. Ontanon, G. Synnaeve, A. Uriarte, F. Richoux, D. Churchill, and M. Preuss, "A Survey of Real-Time Strategy Game AI Research and Competition in StarCraft", *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 5, no. 4, pp. 293–311, Dec. 2013, ISSN: 1943-068X. DOI: 10.1109/TCIAIG.2013.2286295. [Online]. Available: http://ieeexplore.ieee.org/document/6637024/.

[3] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, and D. Hassabis, "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play.", *Science (New York, N.Y.)*, vol. 362, no. 6419, pp. 1140–1144, Dec. 2018, ISSN: 1095-9203. DOI: 10.1126/science.aar6404. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/30523106.

[4] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates", in *2017 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2017, pp. 3389–3396.

[5] W. H. Guss, C. Codel, K. Hofmann, B. Houghton, N. Kuno, S. Milani, S. Mohanty, D. P. Liebana, R. Salakhutdinov, N. Topin, M. Veloso, and P. Wang, "The MineRL Competition on Sample Efficient Reinforcement Learning using Human Priors", Apr. 2019. [Online]. Available: http://arxiv.org/abs/1904.10079.

[6] L.-J. Lin, "Self-improving reactive agents based on reinforcement learning, planning and teaching", *Machine learning*, vol. 8, no. 3-4, pp. 293–321, 1992.

[7] H. Van Hasselt, A. Guez, and D. Silver, "Deep Reinforcement Learning with Double Q-learning", Tech. Rep. [Online]. Available: www.aaai.org.

[8] G. Lample and D. S. Chaplot, "Playing fps games with deep reinforcement learning", in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[9] M. Kempka, M. Wydmuch, G. Runc, J. Toczek, and W. Jaskowski, "ViZDoom: A Doom-based AI research platform for visual reinforcement learning", in *2016 IEEE Conference on Computational Intelligence and Games (CIG)*, IEEE, Sep. 2016, pp. 1–8, ISBN: 978-1-5090-1883-3. DOI: 10.1109/CIG.2016.7860433. [Online]. Available: http://ieeexplore.ieee.org/document/7860433/.

[10] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, "The arcade learning environment: An evaluation platform for general agents", *Journal of Artificial Intelligence Research*, vol. 47, pp. 253–279, 2013.

[11] 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.

[12] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized Experience Replay", Nov. 2015. [Online]. Available: http://arxiv.org/abs/1511.05952.

[13] Z. Wang, V. Bapst, N. Heess, V. Mnih, R. Munos, K. Kavukcuoglu, and N. de Freitas, "Sample Efficient Actor-Critic with Experience Replay", Nov. 2016. [Online]. Available: http://arxiv.org/abs/1611.01224.

[14] A. Dosovitskiy and V. Koltun, "Learning to Act by Predicting the Future", Nov. 2016. [Online]. Available: http://arxiv.org/abs/1611.01779.