INF8953DE Project Report

Vijaya Lakshmi Kuruba, Emile Dimas, Pavithra

TOTAL POINTS

90 / 90

QUESTION 1

1 Project report 60 / 60

✓ **- 0 pts** **Correct**

- **- 4.8 pts** See comments for details.
- **- 4.2 pts** Click here to replace this description.
- **- 11.4 pts** Click here to replace this description.
- **- 4.8 pts** Click here to replace this description.
- **- 16.2 pts** Click here to replace this description.
- **- 10.8 pts** Click here to replace this description.
- **- 6.6 pts** See comments for details.
- **- 5.4 pts** Click here to replace this description.
- **- 3 pts** See comments for details.
- **- 8.4 pts** Click here to replace this description.
- **- 16.2 pts** Click here to replace this description.
- **- 0.6 pts** Click here to replace this description.
- **- 10.2 pts** See comments for details.
- **- 7.2 pts** See comments for details.
- **- 9 pts** see comments
- **- 3 pts** see comments
- **- 13.2 pts** See comments
- **- 13 pts** See comments
- **- 7.8 pts** Click here to replace this description.
- **- 15 pts** Click here to replace this description.
- **- 1.2 pts** See comments for details.
- **- 6.6 pts** See comments for details.
- **- 0.6 pts** See comments for details
- **- 1.8 pts** Click here to replace this description.
- **- 6 pts** See comments for details.

Rubric:
Paper format + quality of writing: 20/20%
Abstract: experience replay was proposed in the 90s by Lin et al. (as referenced in your introduction section).

Implementation quality/difficulty (How hard was it to code it?): 30/30%
Shows proper and deep understanding of every aspect of the existing algorithm: 10/10%
Experimental setup and rigor (are the experiments proposed logical and purposeful? Are they well set up and executed to observe the desired effect(s)?): 20/20%
Meaningful conclusions that provide new insight on the original work: 20/20%

Overall score: 100/100% = 60/60

QUESTION 2

2 Presentation 30 / 30

✓ **+ 30 pts** .

- **+ 24 pts** .
- **+ 27 pts** .
- **+ 25 pts** .
- **+ 22 pts** Click here to replace this description.
- **+ 28 pts** Click here to replace this description.

BEYOND PRIORITIZED EXPERIENCE REPLAY

Pavithra Parthasarathy
pavithra.parthasarathy-rajasekar@mila.quebec

Vijaya Lakshmi Kuruba
lakshmiv@mila.quebec

Emile Dimas
emile.dimas@mila.quebec

ABSTRACT

Experience replay , which was introduced in the DQN paper (Mnih et al., 2013) (Mnih et al., 2015) is a feature that gives online reinforcement learning agents the ability to remember and to reuse experiences from the past. In this ablation study, we explored how prioritizing experiences allows to replay important transitions more frequently, and therefore learn more efficiently. We studied how different original variants of the Prioritized Experience Replay such as the priority based and rank based approaches fare in the Cartpole v0 environment. Moreover, we tested the performance of new variants of the PER in the same environment. We also tried more complex environments such as Lunarlander v2 to stress test our models. Comparisons between the original implementation and our additions were inconclusive.

1 INTRODUCTION AND PROBLEM DEFINITION

Deep Q-Network (DQN) (Mnih et al., 2013) (Mnih et al., 2015) was one of the main innovations in the deep Reinforcement Learning community. This algorithm introduced many features to solve big issues faced in the online RL community until then, namely strongly correlated updates that don't respect the i.i.d. assumption of stochastic gradient descent and the rapid forgetting of very rare experiences. One of the additions brought by DQN was the Experience Replay (ER) (Lin, 2004). Prior to ER, RL agents experienced transitions only once and then discarded it. ER is a data structure of size N used by off-policy algorithms that stores experienced transitions. During training, the agent samples uniformly a mini batch of experiences from it and update the agent parameters consequently. ER has two main benefits: First, the uniform sampling causes the agent to process a more diverse mini batch of transitions and thus remove the correlation between the samples. Second, it keeps the agent from forgetting old transitions that would be useful later in the episode. With ER, rare experiences have the same probability of being used for updates than non rare transitions which means that they will be replayed more often compared to an online algorithm. Empirical research has shown robustly that algorithms using experience replay converges faster and to better results.

However, ER still has some drawbacks. In fact, in sparse reward settings, ER has a lot of difficulty to converge since all transitions have the same probability of being chosen. In a sparse reward setting, useless transitions are overwhelmingly more numerous and therefore the probability of being used for the model update will be larger than the probability of using a useful transition. Prioritized Experience Replay (PER) introduced by (Schaul et al., 2015) solves this issue. The key idea of PER is that an RL agent can learn more effectively from some transitions than from others. Figure 1 is an example of environmnent with rare reward setting in which prioritization is key for solving the problem more efficiently.

In this project, we push the boundaries of the PER paper further by asking the question: Is it possible to find better proxy measures that yields better results? In the first part of this report, we give a detailed summary of the PER mechanism and explain our motivation to do this ablation study. In the second paragraph, we present a brief overview of the related work. Then we introduce in detail the different mechanisms we used with a discussion on the obtained results and the

limitations faced. Finally, we conclude the report. A link to our github repo can be found at the end of the report followed by a small paragraph describing the contribution of each members.

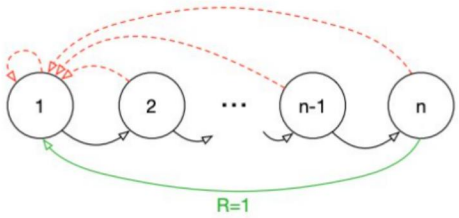


Figure 1: Blind Cliff Walking Setup

2 Background and Motivation

As mentioned above, PER liberates the RL agent from using transitions uniformly. It affects a sampling probability proportional to the priority of the transition. This idea was inspired by the use of priority in previous algorithms like Prioritized sweeping. The obvious question that arises is how can the priorities be computed. Theoretically, the priority should be high for transitions that are the most useful in the learning process. However, there is no quantitative value that directly measures the degree of usefulness of an arbitrary transition. In the PER paper (Schaul et al., 2015), the authors choose to approximate this metric by a proxy variable, this variable being the magnitude of the temporal difference (TD) error which is a measure of how "surprising" and "unexpected" a transition is.

In the PER paper, the authors introduce two types of priority:

1. Proportional prioritization (proportion based): $p_i = |\delta_i|$
2. Rank based prioritization (rank based): $p_i = \frac{1}{rank(|\delta_i|)}$

In the case where transitions have a low TD error on the first visit, it may not be replayed for a long time. Also, in a greedy prioritization setting, the buffer is very sensitive to noise and that effect can be amplified by the bootstrapping. Finally, when using function approximation TD errors tend to decrease very slowly which will result in the replay of the same transition over and over before switching to a new one. For all the reasons mentioned above, the authors of PER introduce a stochastic sampling where the probabilities are computed as follows:

$$P(i) = \frac{p_i^\alpha}{\sum_k p_k^\alpha}$$

Finally, since PER changes the distribution of the transitions in an unknown way, the solution is biased and can be corrected by using importance sampling weights computed in the following way:

$$w_i = (\frac{1}{N} \cdot \frac{1}{P(i)})^\beta$$

The β parameter is annealed from β_0 (hyperparameter) to 1. The weights are normalized by dividing the weights by the biggest one.

In some circumstances, the magnitude of the TD error can be a poor proxy measure of the priority. For example, when the rewards are noisy, the TD errors are noisy. This motivates the study presented in this report.

3 Related Works

Our ablation study is based on the paper Prioritized Experience Replay (Schaul et al., 2015). This paper combined prioritized replay algorithm into a full-scale reinforcement learning agent, based on

the state-of-the-art Double NDQ algorithm (van Hasselt Hado et al., 2016). It is an improvement over Experience replay (Lin, 2004) which was demonstarted in Deep Q-Network (DQN) algorithm (Mnih et al., 2013) (Mnih et al., 2015).

4 Detailed discussion of the Research/Analysis Questions studied

Our first step was to replicate the original algorithm in the OpenAI Gym - CartPole-v0 environment using the proportional based and rank based approaches.We then implemented the different variants listed below:

- Replacing the TD error with its derivative.
- Increasing priorities of transitions that have not been replayed for a while.
- Treating positive TD error transitions and negative ones separately.
- Using a hybrid approach where 2 samples are sampled using different priorities.

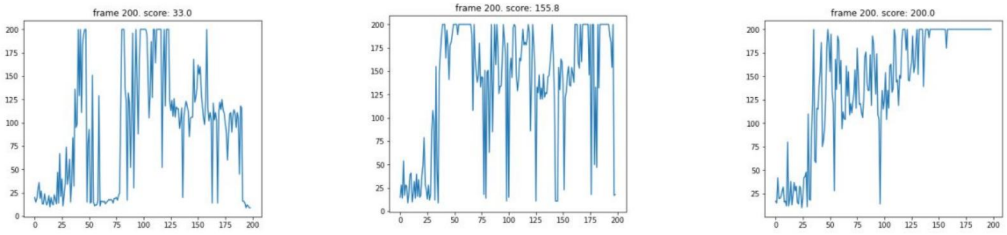
Finally, we run the following list of experiments in CartPole v0 and LunarLander v2 environments:

- Different initial maximum priority
- Original PER with different replay buffer memory size
- Proportion-based vs rank based vs hybrid approach
- Staleness bonus
- Positive bonus
- Derivative of TD error

4.1 Effect of Memory Size on Prioritization

In the paper (Jiang et al., 2020), the authors show that the performance of an agent is not a monotonic function of the memory size when using a standard replay buffer. We wanted to test if it was the same case when introducing priority sampling.

We tried different memory sizes to see how the performance is impacted. We can see that the larger the memory, the better the performance gets. Also, the variance is reduced. This is because we use a FIFO based implementation. Therefore, we have a larger number of transitions stored in the memory from which we can sample. The older transitions with larger TD errors are not flushed out of the queue. Hence, stability. However, When the memory size is too small, the learning process is more likely to incur bad learning (weights updated in the wrong direction) because of the limited memory capacity. Since buffers use a FIFO approach, the transitions presented in the memory will show a high degree of correlation. When we increase the size of the memory, the overshooting effect (bad learning) decreases. We did not have a lot of computation power to test for extremely large memory size to see if it has negative impact on learning.



(a) Memory size: 100 (b) Memory size: 1000 (c) Memory size: 10000

Figure 2: Performance for proportion-based PER with different memory size(Cartpole v0)

4.2 Higher Initial Priority for recently visited transitions

During training, transitions encountered by the agent during its interaction with the environment are not used by the model before a specific number of iterations. Therefore, for implementation efficiency, the agent stores a transition in the buffer without computing the TD error when it first encounters it. Thus, it is important to assign a maximal priority to all the transitions encountered the first time, in order to guarantee that all experiences are seen at least once. This maximal priority is a hyperparameter and its value can greatly affect the convergence of the algorithm. This feature implicitly gives higher priority to recently visited transactions.

The figure below shows the effect of different initialization values. We can see that the best results are obtained with the initial priority value of 1. The learning worsens monotonically with higher values of initial priority. This can be explained by the fact that the TD errors' magnitude which is used as priority is way smaller than the maximum priority (L1 smooth loss). Therefore, when computing the sampling probabilities (using the softmax over priorities), the probabilities corresponding to the recently encountered transitions will be overwhelmingly higher. Thus, the sampling process will resemble an online algorithm and the transitions used for learning will present some correlation which explains the poor results.

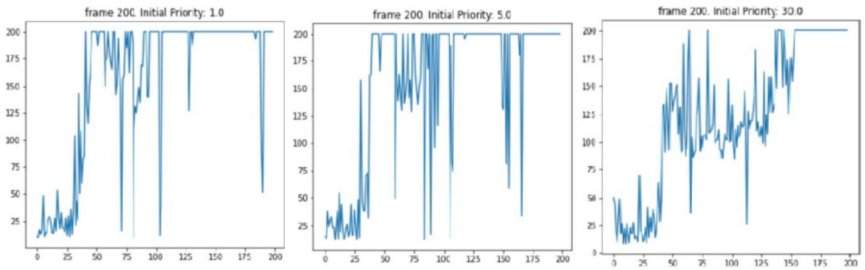


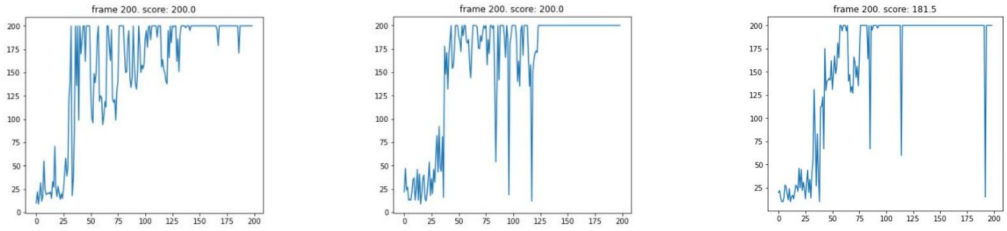
Figure 3: Initializing TD Errors at different values in proportion-based PER

4.3 Introducing stochasticity in the sampling process

4.3.1 Hybrid Prioritization

The hybrid approach consists of combining transitions sampled according to proportional based prioritization and rank based priority in a minibatch. The intuition behind this approach is to introduce additonal diversity, which is important to prevent overfitting, premature convergence and increase exploration. The hybrid method may also combine the strength of proportional based and rank based. In fact, theoretically, rank based is more robust and presents less variance since the use of the rank doesn't show huge differences of magnitude unlike the propotional based method. It therefore removes outliers. However, while rank based method presents theoretically backed arguments, in practice it is slower than proportional based since it is necessary to sort the entire buffer. Sorting has a time complexity of $O(n^2)$.

The results presented in the figure indicate that the hybrid method converges slower than the other two, but the variance shown in the graph is low (compared to others). We speculate that by combining the two methods (proportional and rank based) the agent has a better exploration strategy.



(a) proportion-based (b) rank based (c) hybrid

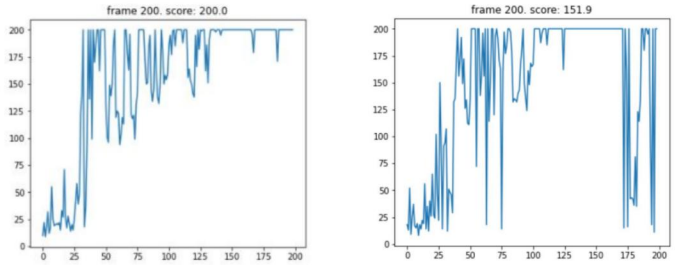
Figure 4: Performance for the 3 different methods of PER (proportion vs rank vs hybrid in Cartpole v0)

4.4 Exploring other proxies for Priority Measure instead of TD-error

4.4.1 Prioritization using the derivative of the TD Error

As mentioned previously the TD error measures how surprising a transition is. However, it doesn't consider the stochasticity of the rewards and the limitations brought forward by the use of function approximation. Therefore, it is problematic when transitions are unlearnable. One solution is the use of the derivative of the TD error. In our implementation, the TD error derivative is approximated by the difference of TD error when a transition is replayed. This will increase the priority whenever consecutive error signs match and decrease when there is difference in sign.

In our experiments for cartpolev0 environment from Figure 5, we see that it converges slower than if we use TD Error. Moreover, we observe high variance in the rewards. This can be explained by the complex implementation of the derivative. In fact, the derivative is not directly available, and it is necessary to sample other transitions in the meantime which increases the variance of the derivative measure. One way to solve this problem would be to train another network only on the derivative computed but that would add a lot of complexity and we don't think it is worth the effort.



(a) Without differential b) With differential

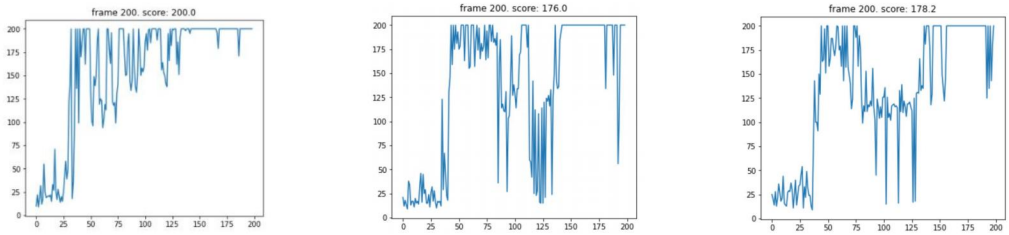
Figure 5: Performance for proportion-based PER without and with differential method (Cartpole v0)

4.4.2 Treating Positive TD error transitions and negative ones separately

An asymmetry in replay frequency was observed in rat studies (Singer et al., 2009).Based on this idea we conducted the following experiment:

We modulated prioritization by not treating positive TD errors the same as the negative ones. We introduce an asymmetry and prioritize positive TD errors over the negative ones of equal magnitude, considering the fact that positive TD errors are more informative than negative ones. Inspired by the penalty used for the staleness bonus (4.4.3) , we use the same technique in this case. In fact, we increase the priority of transitions with positive TD errors by adding an ϵ.

As seen from plot, this positive boosting added varinace. In function approximation setting, all states are interrelated as the weights are shared. Biasing towards states with positive TD errors is a disadvantage for the other states. It is evident from the plots that this experiment doesn't help our learning process.



(a) Positive boosting: 0 (b) Positive boosting: 0.1 (c) Positive boosting: 0.0001

Figure 6: Performance for proportion-based PER with different Positive penalty coefficients (Cartpole v0)

4.4.3 Introducing staleness bonus

Priorities that have not been sampled for a while might get flushed out of the buffer if the memory is full. Some of these priorities might have useful information even though the first times the agent assigned to them a low priority. This may be due to the stochasticity of the environment and its reward sturcture. The key idea here is motivated by the paper Prioritized Level Replay (Schaul et al., 2015). However, in our case we try a much simpler version of their implementation. We increase the priority of elements not visited for a while by subtracting the weighted (by the staleness coefficient) global step count to the TD error when the transition is encountered.

We ran the experiments with different staleness coefficients. When the staleness coefficient is close to the range of TD errors' magnitude (we use L1 smooth loss), the convergence is much better. However, if the weighted global step count magnitude is much higher than the TD error, it boosts the over all priority for all transitions and tends to act as a standard replay buffer.

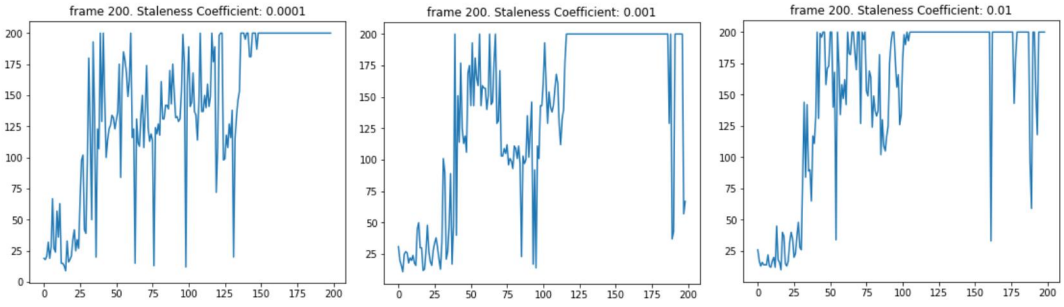


Figure 7: Comparison of returns during training at different iterations for different staleness coefficients in proportion-based method

4.5 Study of experiments on Lunar Lander V2 Environment

We have run our experiments on Lunar Lander V2 open gym environment and the results have been reported here. OpenAI Gym's Lunar Lander environment in one of 4 discrete actions at each time step returns a state in an 8-dimensional continuous state space along with a reward.

We can observe from plots that the rank based method converges slower than the other two methods and we have a smoother transition for hybrid priority. However, the results are not optimal in the hybrid setup. We think this is due to the fact that transitions coming from two different distributions might add noise to the gradient vector update (counter act).

The environment is a bit more complicated than cartpole, but still we see that the agent is converging. There is a lot of noise in the results since the environment is stochastic. Experiments on more complicated environments might be needed for more conclusive results.

Plots for the other ablation studies are added in Appendix.

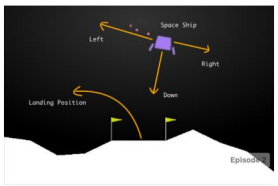
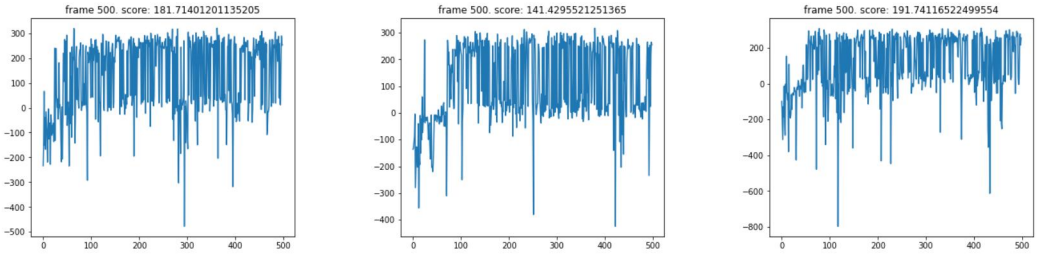


Figure 8: LunarLander



(a) Proportion Based (b) Rank Based (c) Hybrid priority

Figure 9: Performance of different PER options in Lunar Landerv2

5 Limitations

We have 2 limitations in our study.

- The environments we used for our tests were Cartpole (deterministic environment) and LunarLander (stochastic environment). These seem to be very simple environments for our experiments. So, the agent is always able to converge for all our studies. Only the inital convergence varies.
- The runs were not averaged over multiple iterations. So, the rewards are noisy and it is hard to conclude with certainty.

6 Conclusion

In this ablation study, we explored different variants of the Prioritized Experience Replay method. Namely, we tried a hybrid approach which combines proportional and rank based methods. We also experimented by replacing the magnitude of the TD error with its derivative. Finally, we tried to incorporate staleness bonuses and positive boosting to certain types of transitions.
In general, we observe that in the test phase, agents mostly act optimally even though the training curves are mostly noisy. Since our environments are very simple, our experiments don't present cases of major failure. Therefore our results are inconclusive and it is necessary to explore more complex environments(i.e: Atari games) to study the effect of our variants.

7 Links to Code

The link to the the github repository can be found here.

8 Contributions by each Team Member

The workload was split equally among all the team members.

- Rank Based Prioritization, Proportional Prioritization, Hybrid prioritization implementation and study of experiments on them done by **Emile Dimas**.
- Staleness Bonus, Effect of Initial Priorities and Prioritization using derivative of the TD Error implemenation and study of experiments done by **Pavithra Parthasarathy**.
- Treating Positive and Negative TD errors differently implementation and set up and study of all the experiments on Lunar Lander environment by **Vijaya Lakshmi Kuruba**

References

Minqi Jiang, Edward Grefenstette, and Tim Rocktäschel. Prioritized level replay. *CoRR*, abs/2010.03934, 2020. URL https://arxiv.org/abs/2010.03934.

Longxin Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine Learning*, 8:293–321, 2004.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning, 2013.

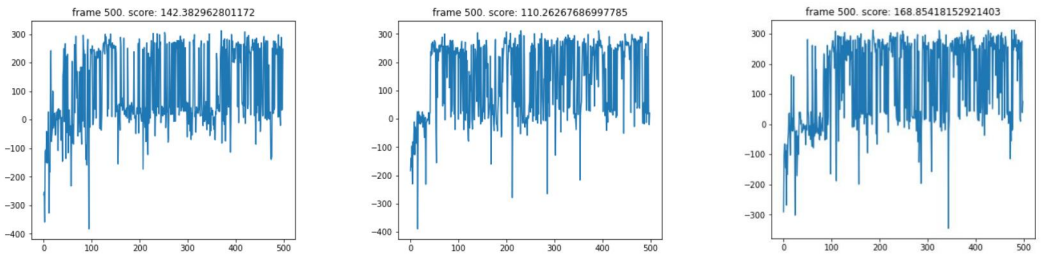
Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin A. Riedmiller, Andreas Fidjeland, Georg Ostrovski, Stig Petersen, Charlie Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

Annabelle C Singer, Annabelle C Singer, and Loren M Frank. Rewarded outcomes enhance reactivation of experience in the hippocampus. *Neuron*, 64(6):910—921, December 2009. ISSN 0896-6273. doi: 10.1016/j.neuron.2009.11.016. URL https://europepmc.org/articles/PMC2807414.

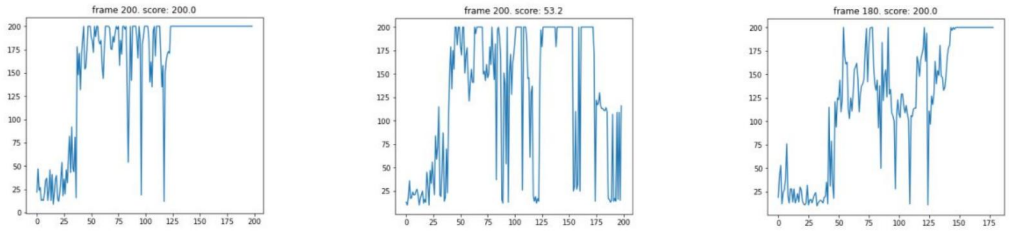
van Hasselt Hado, Guez Arthur, and Silver David. Deep reinforcement learning with double q-learning, 2016.

A Appendix



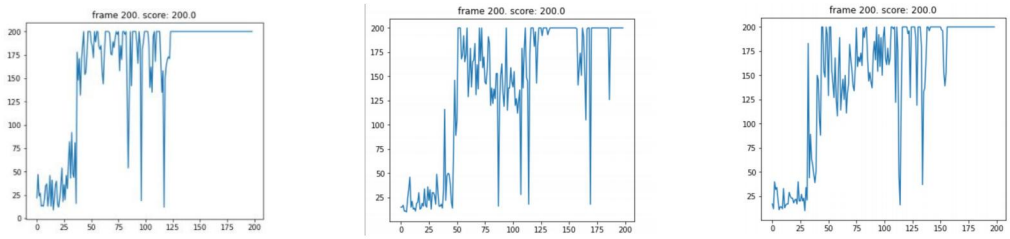
(a) Staleness Coefficient (b) TD Diffrential (c) Positive TD boosting

Figure 10: Performance of different ablation studies of Lunar Lander V2



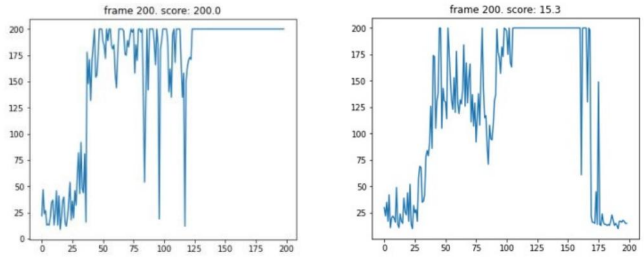
(a) Staleness Coeff: 0 (b) Staleness Coeff: 0.1 (c) Staleness Coeff: 0.0001

Figure 11: Performance for rank based PER with different stalness Coeff (Cartpole v0)



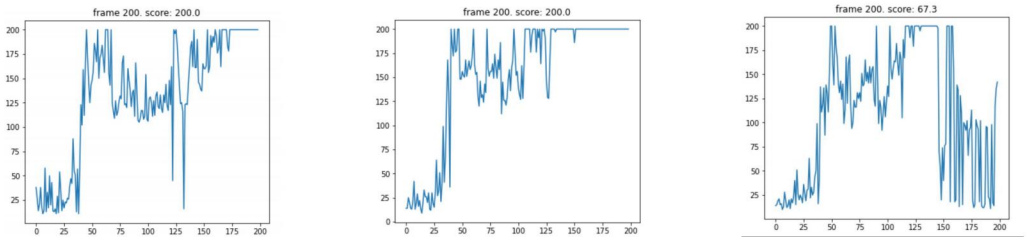
(a) Positive boosting: 0 (b) Positive boosting: 0.1 (c) Positive boosting: 0.0001

Figure 12: Performance for rank based PER with different positive penalty coefficient (Cartpole v0)



(a) Without differential (b)With differential

Figure 13: Performance for rank based PER without and with differential method (Cartpole v0)



(a) priority based (b) rank based (c) hybrid

Figure 14: Performance for the 3 different types of PER (proportion-based, rank based and hybrid approach combining positive penalty and staleness)(Cartpole v0)

1 Project report 60 / 60

✓ **- 0 pts Correct**

- 4.8 pts See comments for details.

- 4.2 pts Click here to replace this description.

- 11.4 pts Click here to replace this description.

- 4.8 pts Click here to replace this description.

- 16.2 pts Click here to replace this description.

- 10.8 pts Click here to replace this description.

- 6.6 pts See comments for details.

- 5.4 pts Click here to replace this description.

- 3 pts See comments for details.

- 8.4 pts Click here to replace this description.

- 16.2 pts Click here to replace this description.

- 0.6 pts Click here to replace this description.

- 10.2 pts See comments for details.

- 7.2 pts See comments for details.

- 9 pts see comments

- 3 pts see comments

- 13.2 pts See comments

- 13 pts See comments

- 7.8 pts Click here to replace this description.

- 15 pts Click here to replace this description.

- 1.2 pts See comments for details.

- 6.6 pts See comments for details.

- 0.6 pts See comments for details

- 1.8 pts Click here to replace this description.

- 6 pts See comments for details.

Rubric:
Paper format + quality of writing: 20/20%
Abstract: experience replay was proposed in the 90s by Lin et al. (as referenced in your introduction section).

Implementation quality/difficulty (How hard was it to code it?): 30/30%
Shows proper and deep understanding of every aspect of the existing algorithm: 10/10%
Experimental setup and rigor (are the experiments proposed logical and purposeful? Are they well set up and executed to observe the desired effect(s)?): 20/20%
Meaningful conclusions that provide new insight on the original work: 20/20%

Overall score: 100/100% = 60/60

2 Presentation 30 / 30

✓ **+ 30 pts** .

+ 24 pts .

+ 27 pts .

+ 25 pts .

+ 22 pts Click here to replace this description.

+ 28 pts Click here to replace this description.