Prioritizing Samples in Reinforcement Learning with Reducible Loss

Anonymous Author(s)
Affiliation
Address
email

A Implementation Details

We build our experiments on top of existing implementations of SAC, DQN and Rainbow. For the DeepMind Control Suite experiments, we modify Yarats and Kostrikov [2020], adding a prioritized replay buffer and the ReLo version. We use an open source implementation of Rainbow[1] for the Arcade Learning Environment and the DQN implementation from the MinAtar authors Young and Tian [2019]. Aside from the collected frames and number of seeds, we have not modified any of the hyper-parameters from these original implementations. The hyper-parameters as well as hardware and software used are given in Table 1.

Table 1: Hyper-Parameters of all experiments

Environments	Algorithm	Algorithm Parameters	Hardware & Software
ALE	Rainbow	Frames = 2×10^6 seeds = 5 Remaining hyper-parameters same as Hessel et al. [2017]	Hardware- CPU: 6 Intel Gold 6148 Skylake GPU: 1 NVidia V100 RAM: 32 GB Software- Pytorch: 1.10.0 Python: 3.8
DeepMind Control Suite	SAC	Frames = 1×10^6 seeds = 5 Remaining hyper-parameters same as Haarnoja et al. [2018]	Hardware- CPU: 6 Intel Gold 6148 Skylake GPU: 1 NVidia V100 RAM: 32 GB Software- Pytorch: 1.10.0 Python: 3.8
MinAtar	DQN	Frames = 5×10^6 seeds = 5 Remaining hyper-parameters same as Mnih et al. [2015]	Hardware- CPU: 6 Intel Gold 6148 Skylake GPU: 1 NVidia V100 RAM: 32 GB Software- Pytorch: 1.10.0 Python: 3.8

[1] https://github.com/Kaixhin/Rainbow

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

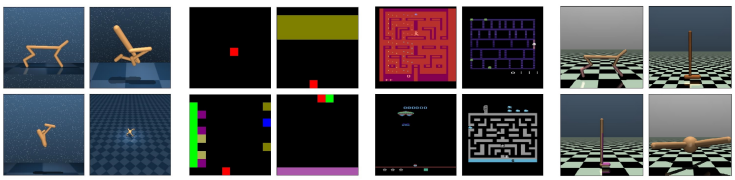

Figure 1: Visualization of a few environments from each benchmark. Left to right: DeepMind Control Suite, MinAtar, Arcade Learning Environment

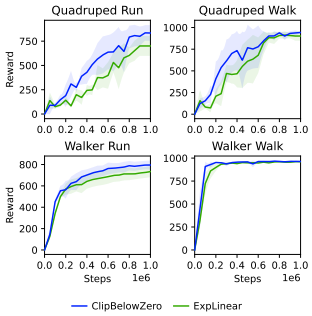

Figure 2: Comparison of different mapping functions from ReLo to p_i on a subset of environments from the DMC benchmark. Performance is evaluated over 3 seeds.

B Mapping Functions for ReLo

Prioritized experience replay buffers expect the priorities assigned to data points to be non-negative. While the MSE version of the TD error used in PER satisfies this constraint, ReLo does not. Therefore, there must be a non-negative, monotonically increasing mapping from ReLo to p_i. In our main experiments, we clipped negative ReLo values to zero. Another mapping we tried was to set $p_i = e^{ReLo}$, in which case the probability of sampling a data point P_i, from Eq. 6, corresponds to the softmax over ReLo scores. However, for this choice the priority would explode if the ReLo crossed values above 40 which happened occasionally during the initial stages of learning in Rainbow. The second mapping function candidate was exponential when ReLo is negative and linear otherwise, that is,

$$f_{ExpLinear} = \begin{cases} e^{\text{ReLo}} & \text{if ReLo} < 0 \\ \text{ReLo} + 1 & \text{otherwise} \end{cases}$$

The linear part is shifted so that the mapping is smooth around ReLo = 0. As shown in Fig. 2, $f_{\text{ExpLinear}}$ performs worse compared to just clipping negative ReLo values to zero. When the ReLo values during training are analysed, we observe that the average of ReLo values (before the mapping) tends to be positive, so clipping does not lead to a large loss in information.

C Extended Related Work

C.1 Prioritization Schemes

Prioritization strategies have been leading to important improvements in sample efficiency. Sinha et al. [2020] proposes an approach that re-weights experiences based on their likelihood under the stationary distribution of the current policy in order to ensure small approximation errors on the value function of recurring seen states. Lahire et al. [2021] introduces the Large Batch Experience Replay (LaBER) to overcome the issue of the outdated priorities of PER and its hyperparameter sensitivity by employing an importance sampling view for estimating gradients. LaBER first samples a large batch from the replay buffer then computes the gradient norms and finally down-samples it to a mini-batch of smaller size according to a priority.

Kumar et al. [2020] presents Distribution Correction (DisCor), a form of corrective feedback to make learning dynamics more steady. DisCor computes the optimal distribution and performs a

weighted Bellman update to re-weight data distribution in the replay buffer. Inspired by DisCor, Regret Minimization Experience Replay (ReMERN) Liu et al. [2021] estimates the suboptimality of the Q value with an error network. Yet, Hong et al. [2022] uses Topological Experience Replay (TER) to organize the experience of agents into a graph that tracks the dependency between Q-value of states.

While PER was initially proposed as an addition to DQN-style agents, Hou et al. [2017] have shown that PER can be a useful strategy for improving performance in Deep Deterministic Policy Gradients (DDPG) Lillicrap et al. [2016]. Another recent strategy to improve sample efficiency was to introduce losses from the transition dynamics along with the TD error as the priority Oh et al. [2022]. Although this has shown improvements, it involves additional computational complexity since it also requires learning a reward predictor and transition predictor for the environment. Our proposal does not require training additional networks and hence is similar in computational complexity to PER. This makes it very simple to integrate into any existing algorithm. Wang and Ross [2019] propose an algorithm to dynamically reduce the replay buffer size during training of SAC so that the agent prioritizes recent experience while also ensuring that updates performed using newer data are not overwritten by updates from older data. However, they do not distinguish between points based on learn-ability and only assume that newer data is more useful for the agent to learn.

C.2 Off-Policy Algorithms

Off-policy algorithms are those that can learn a policy by learning from data not generated from the current policy. This improves sample efficiency by reusing data collected by old versions of the policy. This is in contrast to on-policy algorithms such as PPO Schulman et al. [2017], which after collecting a batch of data and training on it, discard those samples and start data collection from scratch. Recent state-of-the-art off-policy algorithms for continuous control include Soft Actor Critic (SAC) Haarnoja et al. [2018] and Twin Delayed DDPG (TD3) Fujimoto et al. [2018]. SAC learns two Q networks together and uses the minimum of the Q values generated by these networks for the Bellman update equation to avoid over estimation bias. The Q target update also includes a term to maximize the entropy of the policy to encourage exploration, a formulation that comes from Maximum Entropy RL Ziebart et al. [2008]. TD3 is a successor to DDPG Lillicrap et al. [2016] which addresses the overestimation bias present in DDPG in a similar fashion to SAC, by learning two Q networks in parallel, which explains the "twin" in the name. It learns an actor network μ following Eq. 4 to compute the maximum over Q values. TD3 proposes that the actor networks be updated at a less frequent interval than the Q networks, which gives rise to the "delayed" name. In discrete control, Rainbow Hessel et al. [2017] combines several previous improvements over DQN, such as Double DQN van Hasselt et al. [2016], PER Schaul et al. [2016], Dueling DQN Wang et al. [2016], Distributional RL Bellemare et al. [2017] and Noisy Nets Fortunato et al. [2018].

D DeepMind Control Suite

We choose 9 environments from the DeepMind Control Suite Tassa et al. [2018] for testing the performance of ReLo on continuous control tasks. Each agent was trained on proprioceptive inputs from the environment for 1M frames with an action repeat of 1. The training curves for the baselines and ReLo are given in Fig. 3.

E OpenAI Gym Environments

We evaluate agents for 1M timesteps on each environment and similar to DM Control, they are trained using proprioceptive inputs from the environment. The hyperparameters for this benchmark are shared with those used for the DM Control Suite experiments.

F MinAtar

We evaluate the baselines against all 5 environments in the MinAtar suite Young and Tian [2019]. A visualization of a few environments from the suite is presented in Fig. 1. Each agent receives the visual observations from the environment and is trained for 5M frames following the evaluation methodology outlined in Young and Tian [2019]. The training curves are given in Fig. 5.

Table 2: Comparison of PER and ReLo on the DMC benchmark

	BASELINE	PER	LABER	RELO
CHEETAH RUN	761.89 ± 112.38	**831.87 ± 38.90**	579.80 ± 60.61	660.29 ± 141.22
FINGER SPIN	966.68 ± 29.40	975.40 ± 6.75	884.46 ± 20.23	**978.78 ± 14.46**
HOPPER HOP	**264.75 ± 37.90**	217.35 ± 113.79	90.85 ± 57.76	247.81 ± 51.01
QUADRUPED RUN	612.67 ± 143.90	496.42 ± 216.01	544.80 ± 41.84	**833.92 ± 81.05**
QUADRUPED WALK	831.92 ± 74.34	766.30 ± 200.86	716.61 ± 270.36	**942.64 ± 9.75**
REACHER EASY	**983.06 ± 2.70**	981.58 ± 6.33	947.20 ± 14.46	979.08 ± 11.02
REACHER HARD	955.08 ± 38.52	935.08 ± 47.94	951.08 ± 6.70	**956.80 ± 38.73**
WALKER RUN	759.13 ± 23.91	755.49 ± 64.35	551.81 ± 58.41	**795.14 ± 42.52**
WALKER WALK	943.67 ± 30.28	957.38 ± 8.24	863.89 ± 112.60	**963.28 ± 5.03**

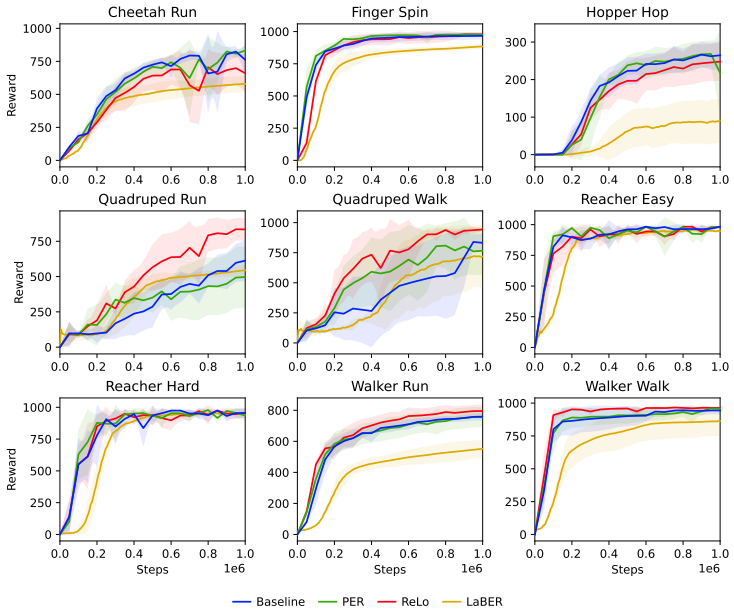

Figure 3: Training curves of environments from the DeepMind Control Suite. Performance is evaluated for 10 episodes over 5 random seeds.

Table 3: Comparison of PER and ReLo on OpenAI Gym environments

	BASELINE	PER	RELO
GYM HALFCHEETAH	9579.60 ± 1331.00	9549.42 ± 917.92	**11590.63 ± 670.36**
GYM HOPPER	**2795.18 ± 659.19**	2175.09 ± 456.11	2527.93 ± 648.61
GYM WALKER	2854.20 ± 623.69	735.16 ± 474.89	**3514.39 ± 676.80**

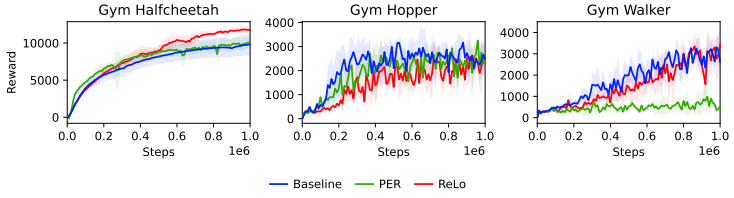

Figure 4: Training curves of environments from the OpenAI Gym benchmark. Performance is evaluated for 10 episodes over 5 random seeds.

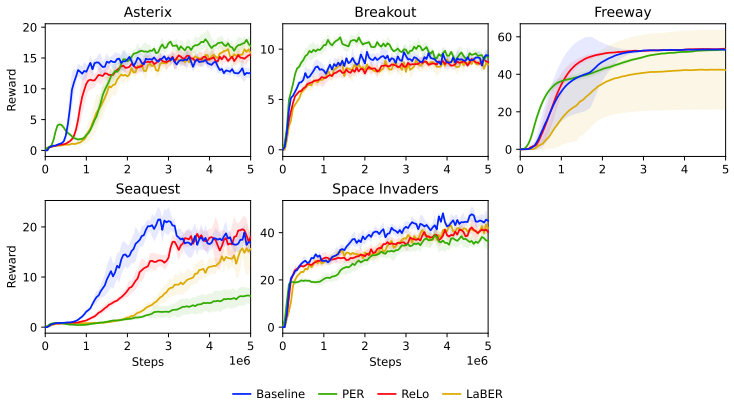

Figure 5: Training curves of environments from the MinAtar benchmark. Performance is evaluated using a running average over the last 1000 episodes over 5 random seeds.

Table 4: Comparison of PER and ReLo on the MinAtar benchmark

	BASELINE	PER	LABER	RELO
ASTERIX	12.54 ± 1.08	$\mathbf{16.16 \pm 1.02}$	15.51 ± 1.11	15.68 ± 0.89
BREAKOUT	$\mathbf{9.36 \pm 0.29}$	8.88 ± 0.72	8.81 ± 0.61	8.98 ± 0.75
FREEWAY	52.80 ± 0.35	52.75 ± 0.22	41.98 ± 20.99	$\mathbf{53.25 \pm 0.37}$
SEAQUEST	16.13 ± 2.88	6.02 ± 1.92	14.63 ± 2.98	$\mathbf{18.13 \pm 1.25}$
SPACE INVADERS	$\mathbf{45.36 \pm 1.65}$	37.36 ± 4.45	43.67 ± 3.17	38.54 ± 2.60

G Arcade Learning Environment

We evaluate agents on a compute-constrained version of the Arcade Learning Environment Bellemare et al. [2013], training each agent for 2M frames. We chose the standard 24 environments from the suite for our evaluation. ReLo is competitive with PER Schaul et al. [2016] in the tested environments. The training curves for the Temporal Difference Error and the rewards are given in Fig. 6 & Fig. 7 respectively.

H Gridworld

We implement a simple GridWorld for the experiments that highlights the drawbacks of PER with TD loss prioritization in Section 5.6. It consists of a 7×7 grid. A visualization of the grid is given in Fig. 8. The start state of the agent, represented by the blue point, is at the top left and the goal state is at the bottom right, represented by the green point. The agent is represented by the black point. The agent gets a reward of $+2$ when it reaches the goal state, but does not receive reward anywhere else other than the stochastic point. The red point marks the state with a corresponding reward uniformly sampled from $[-0.5, 0.5]$. The locations of these points is fixed and does not change during training or evaluation. The state that the agent receives is (x, y) where x and y are the coordinates of the agent's location, $x, y \in [-3, 3]$ and $x, y \in \mathbb{Z}$. The agent has 4 actions, to move up, down, left or right which will deterministically move the agent in that direction by 1 unit. If the agent is at the edge of the grid and takes an action that will move it out of the 7×7 grid, then it remains in the same location.

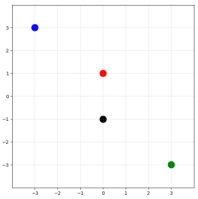

Figure 8: A top down view of the GridWorld. The agent is the black point. It starts at the blue point and the goal state is the green point. The red point represents the location of the stochastic reward.

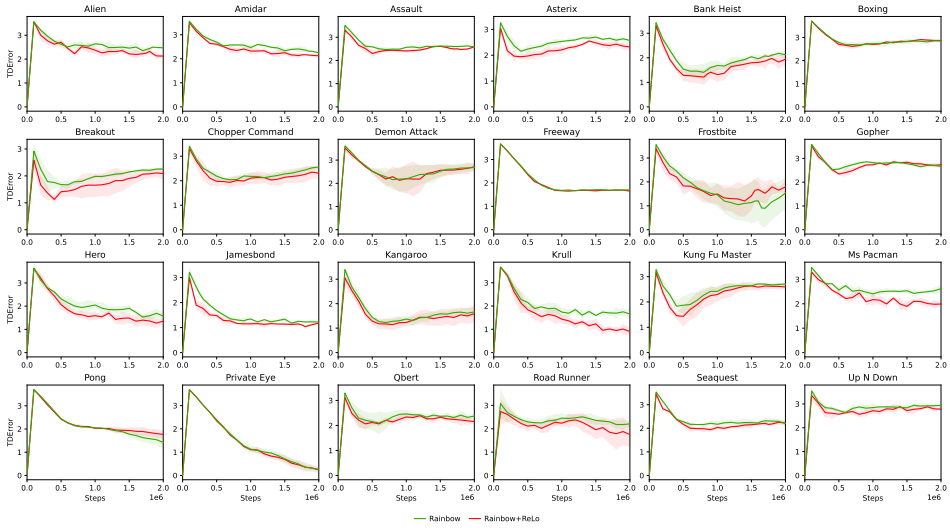

Figure 6: Temporal difference loss curves for Rainbow (with PER) and Rainbow with ReLo. Rainbow with ReLo achieves lower loss compared to PER, showing that ReLo is able to prioritize samples with reducible loss. The dark line represents the mean and the shaded region is the standard deviation over 3 seeds.

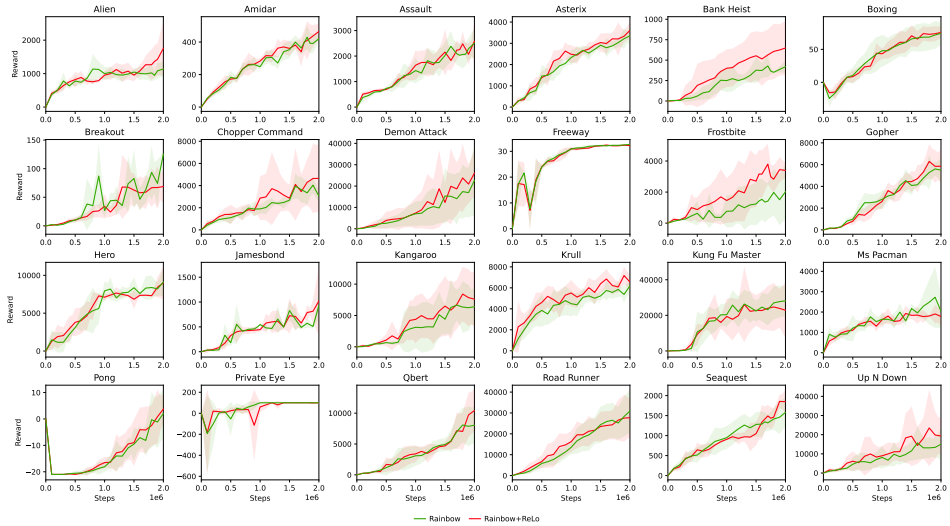

Figure 7: Training curves of 24 environments from the ALE benchmark. Performance is evaluated for 10 episodes over 5 random seeds.

References

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.

Marc G Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *International Conference on Machine Learning*, pages 449–458. PMLR, 2017.

Meire Fortunato, Mohammad Gheshlaghi Azar, Bilal Piot, Jacob Menick, Matteo Hessel, Ian Osband, Alex Graves, Volodymyr Mnih, Rémi Munos, Demis Hassabis, Olivier Pietquin, Charles Blundell, and Shane Legg. Noisy networks for exploration. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL `https://openreview.net/forum?id=rywHCPkAW`.

Table 5: Comparison of Rainbow with PER and Rainbow with ReLo on the ALE benchmark

	RAINBOW	RAINBOW W/ RELO
ALIEN	1278.70 ± 223.14	**1352.90 ± 535.70**
AMIDAR	376.20 ± 81.07	**410.10 ± 85.63**
ASSAULT	2241.78 ± 648.17	**2617.37 ± 555.97**
ASTERIX	3214.50 ± 323.70	**3352.00 ± 431.98**
BANKHEIST	526.80 ± 277.83	**641.80 ± 284.67**
BOXING	**76.65 ± 14.58**	76.21 ± 11.59
BREAKOUT	**82.03 ± 46.10**	68.36 ± 45.39
CHOPPERCOMMAND	2794.00 ± 732.87	**4974.00 ± 2801.93**
DEMONATTACK	25500.50 ± 16311.26	**29294.30 ± 15905.25**
FREEWAY	**32.58 ± 0.31**	32.35 ± 0.25
FROSTBITE	2850.60 ± 1553.34	**3532.30 ± 1270.73**
GOPHER	5336.60 ± 1023.75	**5679.80 ± 1199.68**
HERO	**9907.15 ± 2108.71**	8830.30 ± 1894.51
JAMESBOND	810.00 ± 412.00	**902.00 ± 469.40**
KANGAROO	**8904.00 ± 3879.97**	8091.00 ± 3618.47
KRULL	6553.18 ± 803.15	**6718.67 ± 799.47**
KUNGFUMASTER	**29371.00 ± 8525.69**	23654.00 ± 12360.40
MSPACMAN	**2094.70 ± 614.58**	1755.80 ± 374.32
PONG	3.18 ± 9.02	**5.96 ± 6.45**
PRIVATEEYE	100.00 ± 0.00	100.00 ± 0.00
QBERT	8382.00 ± 2935.16	**10900.25 ± 2704.23**
ROADRUNNER	**29333.00 ± 9465.78**	29222.00 ± 8696.91
SEAQUEST	1377.20 ± 362.57	**1848.80 ± 788.85**
UPNDOWN	17065.40 ± 7637.25	**21241.50 ± 8599.97**

Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1587–1596. PMLR, 10-15 Jul 2018. URL `https://proceedings.mlr.press/v80/fujimoto18a.html`.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 1856–1865. PMLR, 2018. URL `http://proceedings.mlr.press/v80/haarnoja18b.html`.

Matteo Hessel, Joseph Modayil, Hado van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Daniel Horgan, Bilal Piot, Mohammad Gheshlaghi Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. *CoRR*, abs/1710.02298, 2017. URL `http://arxiv.org/abs/1710.02298`.

Zhang-Wei Hong, Tao Chen, Yen-Chen Lin, J. Pajarinen, and Pulkit Agrawal. Topological experience replay. *ICLR*, 2022. doi: 10.48550/arXiv.2203.15845.

Yuenan Hou, Lifeng Liu, Qing Wei, Xudong Xu, and Chunlin Chen. A novel ddpg method with prioritized experience replay. In *Systems, Man, and Cybernetics (SMC), 2017 IEEE International Conference on*, pages 316–321. IEEE, 2017.

Aviral Kumar, Abhishek Gupta, and Sergey Levine. Discor: Corrective feedback in reinforcement learning via distribution correction. *Advances in Neural Information Processing Systems*, 33: 18560–18572, 2020.

Thibault Lahire, M. Geist, and E. Rachelson. Large batch experience replay. *ICML*, 2021.

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In Yoshua

Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. URL http://arxiv.org/abs/1509.02971.

Xu-Hui Liu, Zhenghai Xue, Jing-Cheng Pang, Shengyi Jiang, Feng Xu, and Yang Yu. Regret minimization experience replay in off-policy reinforcement learning. *NEURIPS*, 2021.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles 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(7540):529–533, February 2015. ISSN 00280836. URL http://dx.doi.org/10.1038/nature14236.

Youngmin Oh, Jinwoo Shin, Eunho Yang, and Sung Ju Hwang. Model-augmented prioritized experience replay. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=WuEiafqdy9H.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. URL http://arxiv.org/abs/1511.05952.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv: Arxiv-1707.06347*, 2017.

Samarth Sinha, Jiaming Song, Animesh Garg, and S. Ermon. Experience replay with likelihood-free importance weights. *L4DC*, 2020.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, Timothy Lillicrap, and Martin Riedmiller. Deepmind control suite. *arXiv preprint arXiv: Arxiv-1801.00690*, 2018.

Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In Dale Schuurmans and Michael P. Wellman, editors, *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 2094–2100. AAAI Press, 2016. URL http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12389.

Che Wang and Keith Ross. Boosting soft actor-critic: Emphasizing recent experience without forgetting the past. *arXiv preprint arXiv: Arxiv-1906.04009*, 2019.

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado van Hasselt, Marc Lanctot, and Nando de Freitas. Dueling network architectures for deep reinforcement learning. In Maria-Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1995–2003. JMLR.org, 2016. URL http://proceedings.mlr.press/v48/wangf16.html.

Denis Yarats and Ilya Kostrikov. Soft actor-critic (sac) implementation in pytorch. https://github.com/denisyarats/pytorch_sac, 2020.

Kenny Young and Tian Tian. Minatar: An atari-inspired testbed for thorough and reproducible reinforcement learning experiments. *arXiv preprint arXiv:1903.03176*, 2019.

Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, Anind K Dey, et al. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pages 1433–1438. Chicago, IL, USA, 2008.