Efficient Multi-Horizon Learning for Off-Policy Reinforcement Learning

Raja Farrukh Ali, Nasik Muhammad Nafi, Kevin Duong, William Hsu
Department of Computer Science, Kansas State University
Manhattan, KS 66506, USA
{rfali, nnafi, kevduong, bhsu}@ksu.edu

Abstract

Value estimates at multiple timescales can help create advanced discounting functions and allow agents to form more effective predictive models of their environment. In this work, we investigate learning over multiple horizons concurrently for off-policy deep reinforcement learning using an efficient architecture that combines a deeper network with the crucial components of Rainbow, a popular value-based off-policy algorithm. We use an advantage-based action selection method and our proposed agent learns over multiple horizons simultaneously while using either an exponential or hyperbolic discounting function to estimate the advantage that guides the acting policy. We test our approach on the Procgen benchmark, a collection of procedurally-generated environments, to demonstrate the effectiveness of this approach, and specifically, to evaluate the agent's performance in previously unseen scenarios.

1 Introduction

A reinforcement learning agent learns to maximize the rewards it receives over its expected lifetime. But what if it's estimate of the expected lifetime is biased? While RL algorithms usually plan for a single, fixed, *distant* horizon by setting the discount factor γ to a value closer to 1, the agent may not live long enough as it had previously expected. Thus, it might have fared better by basing its policy on estimates over multiple time horizons; short, intermediate and long. For example, humans make plans for the immediate future (work day), short-term (professional goals) and long-term (retirement savings), but a person's policy (based on valuation of distant rewards) may change drastically because of a terminal diagnosis (curtailment of expected lifetime). Since the goal of an RL agent is to optimize its return (cumulative reward) through a combination of prediction and control, the prediction method used to estimate the value function (estimated return) plays a central role in the agent's performance.

The discount factor γ determines the timescale of the return. When γ is closer to 0, the agent becomes short-sighted and only maximizes near-term reward, whereas when γ reaches closer to 1, the agent values rewards far into the future. It has been shown that a lower discount value early in learning can help policies to converge, although too low a discount can lead to sub-optimal policies [Bertsekas and Tsitsiklis, 1995]. Thus learning a value representation that consider multiple discount factors is important. In deep RL, previous works have attempted to learn a value function at multi-timescale either by concurrently learning value estimate for a fixed set of γs or conditioning the value estimate over γ. All of these methods require multiple fully connected layers at the end of the network to enable γ-specific value prediction and introduce computational overhead compared to the single-valued approach. Off-policy reinforcement learning is generally considered data-efficient in the sense that it can reuse transitions collected from old policies. However, the added computational complexity for multi-horizon learning entails a necessity for an efficient neural network architecture that can enhance the learning process.

In this work, we explore multi-horizon learning from the perspective of enabling performance improvement of off-policy algorithm through deep feature learning and better exploration in diverse settings. We present results on procedurally generated environments, which are designed to test an agent's ability to learn a robust and generalizable policy while also confirming earlier published results on the Arcade Learning Environment benchmark [Bellemare et al., 2013]. Our approach behind the multi-horizon learning is primarily based on Fedus et al. [2019], however, we extend the idea by using an extension of the dueling DQN architecture [Wang et al., 2016] coupled with parametric noise injection in weights and an advantage-based action selection method [Nafi et al., 2022]. We additionally use implementation improvements proposed for Rainbow by Schmidt and Schmied [2021] such as spectral normalization [Miyato et al., 2018] in the residual blocks, mixed precision training, and faster batching with paralleled environments. We evaluate both exponential and hyperbolic discounting functions in the multi-horizon setting and compare and contrast these with learning over a fixed, single-horizon setting, across the Procgen benchmark [Cobbe et al., 2020].

2 Related Work

The need for abstract planning (such as modeling over multiple timescales) is a fundamental problem in AI. Sutton [1995] extend temporal-difference methods to make predictions that are not specific to a single time scale. Studies from psychology and neuroscience show that humans and animals estimate rewards at numerous timescales [Tanaka et al., 2016] and do not discount future rewards exponentially using a single discount factor γ [Mazur, 1997], instead discounting them hyperbolically [Green et al., 1994]. Kurth-Nelson and Redish [2009] proposed the modeling of hyperbolic discounting via distributed exponential discounting and Fedus et al. [2019] extended this formulation to deep reinforcement learning by approximating hyperbolic discounting from exponential discounting while using multi-horizon (multi-timescale) learning as an auxiliary task. Xu et al. [2018] present a meta-learning approach and propose to use γ as a learn-able parameter of their network. Sherstan et al. [2020] propose Γ-nets to predict value-function for any discount factor γ by using timescale as an input to the value estimator. Nafi et al. [2022] propose hyperbolic discounting-based advantage estimation for generalizable policy learning.

3 Methodology

We aim to evaluate multi-horizon learning for value-based algorithm by incorporating several existing extensions to the DQN algorithm [Mnih et al., 2015], some of which have not been used in similar prior works (Fedus et al. [2019]) such as the use a deeper network for the function approximation and addition of the dueling networks [Wang et al., 2016] to the architecture. The network predicts Q-values for each discount factor γ, and these Q-values are used for the agent's learning (loss calculation). We use the advantage values from the dueling heads for action selection. We experiment with four methods: 1) single-horizon agent that learns over a fixed horizon with a single γ value and discounts rewards exponentially; multi-horizon agent that simultaneously learns Q-values for n_γ values, but has the choice between acting either using 2) a hyperbolically discounted advantage, or exponentially discounted advantage corresponding to either the 3) smallest gamma value (γ_0) or the 4) largest gamma value (γ_N).

3.1 Network Architecture and Rainbow Extensions

The proposed network architecture is presented in Figure 1. We employ the deeper IMPALA-CNN architecture [Espeholt et al., 2018] with 15 convolutional layers instead of the small 3-layer network (Nature-CNN) used in Fedus et al. [2019] for multi-horizon learning. The residual blocks [He et al., 2016] in the IMPALA-CNN architecture keep the optimization process light while enabling substantially deeper feature learning. We use the Rainbow [Hessel et al., 2018] agent which combines several independent improvements on top of the Deep Q-Learning framework. Rainbow uses six extensions to DQN namely double DQN [van Hasselt et al., 2016], dueling DQN [Wang et al., 2016], noisy nets [Fortunato et al., 2018], Distributional RL (C51) [Bellemare et al., 2017], prioritized experience replay buffer [Schaul et al., 2016] and n-step returns [Sutton, 1988]. We include five of the six; double DQN, dueling DQN, noisy nets, prioritized experience replay buffer and n-step returns; however, we exclude the Distributional RL (C51) component as we trade off implementation complexity with performance benefits especially while evaluating for lesser number of time steps

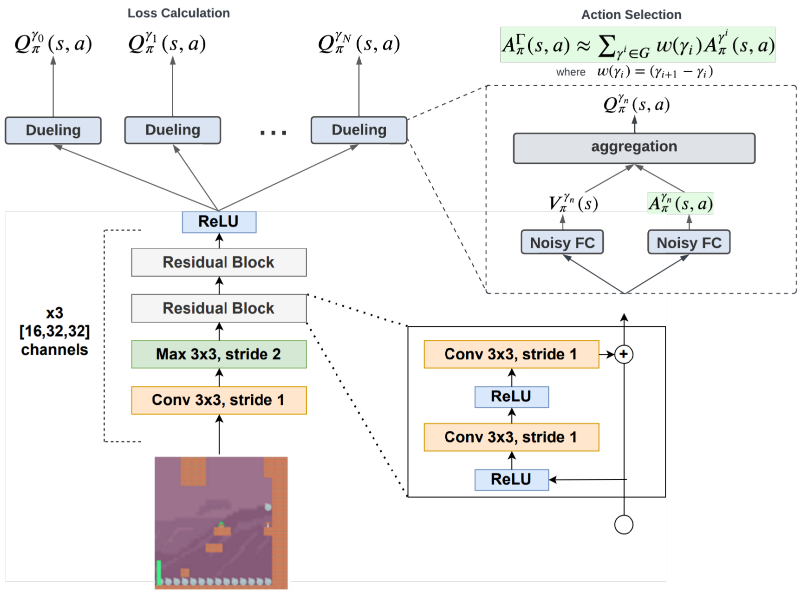


Figure 1: Multi-Horizon network architecture. Output layers predict Q-values for different discount factors using an individual output block for each gamma. Advantages from the dueling head for each gamma value are used for action selection.

(25M). Hessel et al. [2018] show that optimizing the distribution of returns helps in the long run, such as training beyond 40M time steps. Schmidt and Schmied [2021] also note marginal performance improvement with the inclusion of Distributional RL when training for limited (10M) time steps and suggest exclusion of this component. In contrast to Fedus et al. [2019], our approach includes the dueling network and exploration through noisy nets in the multi-horizon learning setting.

3.2 Advantage-based Action Selection

Hyperbolically-discounted advantage estimation has been recently proposed for actor-critic methods [Nafi et al., 2022]. While employing the dueling DQN architecture, we use the advantage from the dueling head for action selection. For the single horizon case (only one output head), the best action is selected based on the highest advantage (argmax over all advantage values). For the multihorizon variant with largest gamma, we use the dueling head corresponding to the largest gamma value (the last head in our implementation), and use the advantage from this head to calculate the best action. As shown by Fedus et al. [2019], we can compute hyperbolic Q-values, $Q_\pi^\Gamma(s,a)$, according to hyperbolic discount factor (k) by considering an infinite set of $Q_\pi^{\gamma^k}(s,a)$ values computed via standard Q-learning. To calculate hyperbolically discounted-advantage for the multihorizon case, we use the advantage values from each of the dueling heads and approximate hyperbolically discounted advantage using a Riemann sum over the discrete interval $G = [\gamma_0, \gamma_1, ..., \gamma_N]$ as:

$$A_\pi^\Gamma(s,a) = \int_0^1 w(\gamma) A_\pi^\gamma(s,a) d\gamma$$

$$A_\pi^\Gamma(s,a) \approx \sum_{\gamma^i \in G} w(\gamma_i) A_\pi^{\gamma^i}(s,a)$$

where weights $w(\gamma_i) = (\gamma_{i+1} - \gamma_i)$.

3.3 The γ Interval

The number of concurrent horizons (n_γ) is an important factor to consider, along with the values of γ that enforce the minimum and maximum horizon for the agent, beyond which the rewards are negligible. We follow the same scheme as suggested by Fedus et al. [2019] for calculating the γ interval, which are the values of γ on which the integral is approximated. These values are produced in a way that taking the exponent of the largest value in this interval with respect to the hyperbolic exponent k approximately equals the γ_{max}, which is the maximum value of horizon we want the agent to use (γ_{max} is set to 0.99). If using a power-method for choosing the γ interval, the base b must satisfy the relation:

$$\gamma_{max} = (1 - b^{n_\gamma})^k$$

where the base b can be solved as $b = exp(ln(1 - \gamma_{max}^{1/k})/n_\gamma)$ which bounds γ_{max} to a stable value. In our experiments, we set n_γ = 5, yielding the γ interval to be [γ = 0.374, 0.608, 0.755, 0.847, 0.904]. For multi-horizon learning with hyperbolic discounting, if we wish to estimate discounting with the form $\Gamma_k(t) = \frac{1}{1+kt}$ where the hyperbolic exponent $k \leq 1.0$, we need to consider the Q-values for γ^k, as these are the actual γ values being learned via Bellman updates. Hence taking an exponent of the last value from the γ interval (0.904) with respect to k (0.1) yields γ_{max} (0.99).

4 Implementation

Using a fast and data-efficient implementation of Rainbow [Schmidt and Schmied, 2021], we implement multi-horizon learning for both hyperbolic discounting and exponential discounting (using advantage corresponding to largest gamma for action selection). Our multi-horizon learning implementation is inspired by Fedus et al. [2019] but differs in a number of ways; we use hyperbolically discounted advantage estimates instead of hyperbolically discounted Q-values for the action selection, the network architecture of Nature CNN is replaced with an IMPALA large (2 channel) network, addition of the dueling networks [Wang et al., 2016] to the architecture, and faster training through hardware improvements (mixed precision training and batched training). The Rainbow implementation in Fedus et al. [2019] used three of the six improvements proposed in Hessel et al. [2018] namely Distributional RL, prioritized replay buffer and n-step returns, whereas our implementation uses five of the six improvements of Rainbow (with the exception of distributional DQN). We build upon the Rainbow implementation of Schmidt and Schmied [2021] to evaluate multi-horizon learning in off-policy reinforcement learning. However, there are some key differences; our evaluation methodology follows pausing training every 250k environment steps to evaluate the agent's current policy for 10 episodes, as opposed to using reloading saved model checkpoints for evaluation. Moreover, we used hyperparameters in line with standard implementations such as Dopamine [Castro et al., 2018] and used environment steps as the standard formulation instead of frames. We use Procgen [Cobbe et al., 2020] to evaluate our approach. The Procgen benchmark is designed to study sample efficiency and generalization in reinforcement learning. The agent is generally trained on a smaller number of levels and expected to perform well in diverse unseen levels. We model the environment as a Partially Observable Markov Decision Process (POMDP) and each level of the game is a sampled POMDP instance.

5 Experiments and Results

In our experiments, the single-horizon agent uses vanilla Rainbow and learns over a fixed horizon with a single γ = 0.99 and discounts rewards exponentially. For multi-horizon, the agent simultaneously learns Q-values for N many γ, but has the choice between acting either using a hyperbolically discounted advantage, $A_\pi^\Gamma(s, a)$, or exponentially discounted advantage, $A_\pi^{\gamma_n}(s, a)$. This exponentially discounted advantage can correspond either to the smallest gamma value (γ_0) making the agent's actions myopic, or the largest gamma value (γ_N), resulting in a far-sighted agent.

5.1 Performance on subset of 4 Procgen environments

In order to test our hypothesis, we initially ran experiments over a subset of 4 Procgen environments. We test the single-horizon (exponential discounting with single gamma) and multi-horizon methods (with largest gamma exponential discounting, smallest gamma exponential discounting and hyperbolic

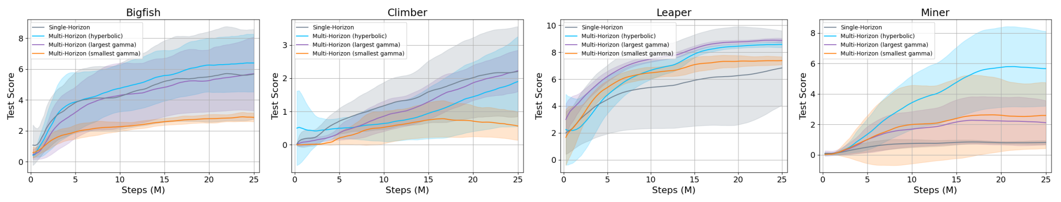


Figure 2: Test performance of Single-Horizon, Multi-Horizon (largest gamma), Multi-Horizon (smallest gamma), and Multi-Horizon (hyperbolic discounting) versions of Rainbow over a subset of four Procgen environments trained over 25M timesteps. Mean and standard deviation are calculated over 5 trials, each with a different seed.

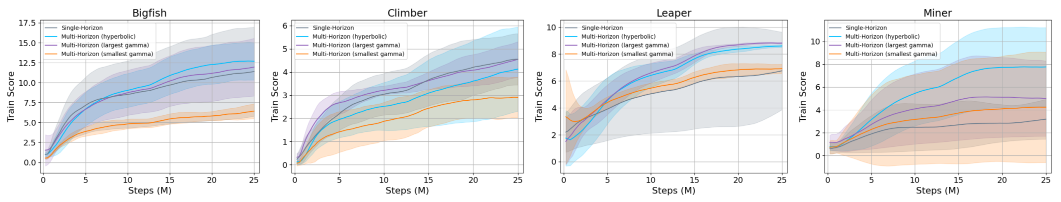


Figure 3: Train performance of Single-Horizon, Multi-Horizon (largest gamma), Multi-Horizon (smallest gamma), and Multi-Horizon (hyperbolic discounting) over four Procgen environments.

discounting) to ascertain the performance of the agent. Figures 2 and 3 shows that the Multi-Horizon (hyperbolic discounting) performed at least as good as Multi-Horizon (largest gamma) and better than both Multi-Horizon (smallest gamma) and Single-Horizon Rainbow variants on three out of the four games, with comparable performance in the other game. Due to the high training cost and poor comparative performance, evaluating Multi-Horizon (smallest gamma) was dropped from subsequent runs.

5.2 Performance on subset of 8 ALE environments

In addition to the Procgen Benchmark, we conduct experiments on a subset of 8 games from the Arcade Learning Environment (ALE) [Bellemare et al., 2013] as done earlier by Fedus et al. [2019]. The results, shown in Figure 4, confirm that multi-horizon learning performs better than the single horizon in most of the environments, which is in line with earlier findings presented by [Fedus et al., 2019], however we use an improved and faster variant of Rainbow (as depicted in the tabular results of Schmidt and Schmied [2021]).

5.3 Test performance on all Procgen environments

Figure 5 shows the test performance of Single-Horizon, Multi-Horizon (largest gamma) and Multi-Horizon (hyperbolic discounting) Rainbow variants on all Procgen games for 25M timesteps. Results indicate that for when an agent is trained on a smaller subset of levels and tested on the full distribution of unseen levels in procedurally generated environments, learning over multiple horizons yields benefits. Multi-horizon methods (either the exponential discounting variant which acts according to the farthest horizon (largest gamma) or the hyperbolically discounted variant, perform better or at-par with the single horizon method (with the exception of Heist whose test scores are much lower than training scores, and Starpilot). When comparing the discounting methods within the multi-horizon setting, we observe no particular method (between exponential and hyperbolic) performing better than the other in all of the games, although we do observe hyperbolic discounting to be marginally better in a majority of the environments. This finding is similar to Fedus et al. [2019] who found multi-horizon learning to be an effective auxiliary task for the agent, irrespective of which discounting function is employed. However, as shown in the next section , hyperbolic discounting does show improved results when compared with exponential discounting with largest γ across the entire Procgen benchmark.

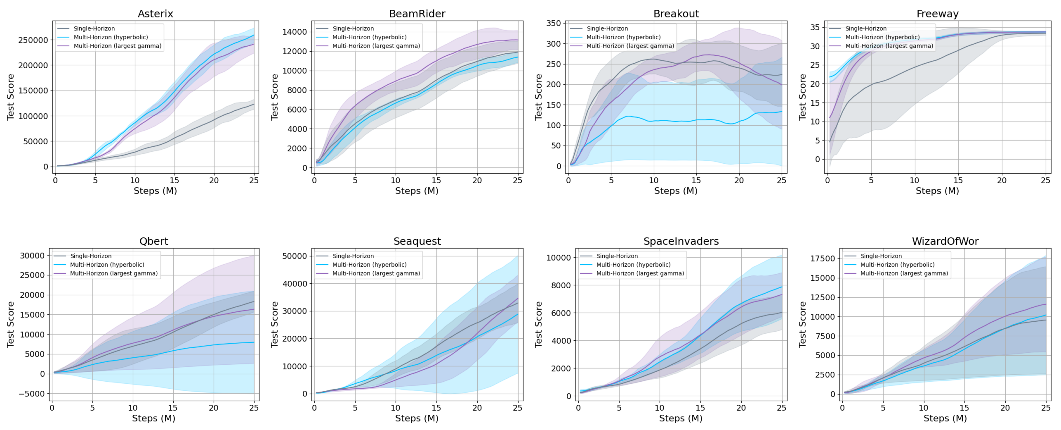


Figure 4: Test performance of Single-Horizon, Multi-Horizon (largest gamma), and Multi-Horizon (hyperbolic discounting) over eight ALE environments. Mean and standard deviation are calculated over 3 trials, each with a different seed.

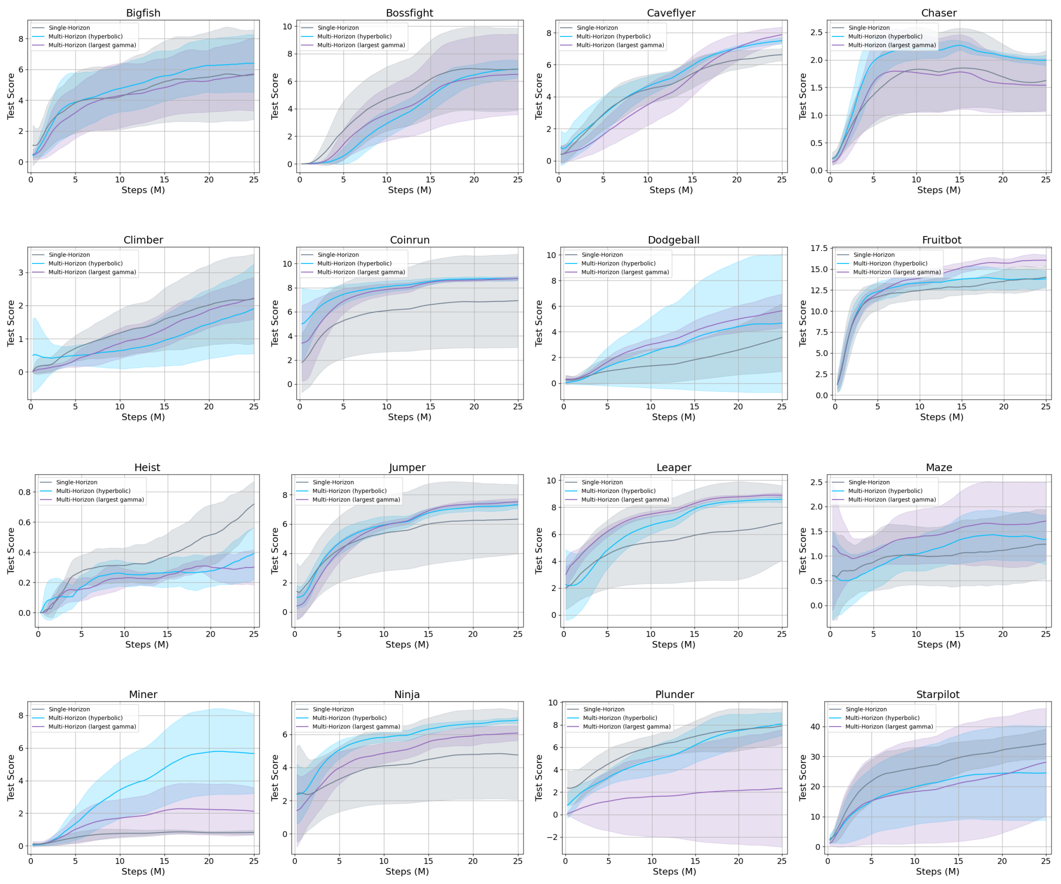


Figure 5: Test performance for Single-Horizon, Multi-Horizon (largest gamma) and Multi-Horizon (hyperbolic) variants of Rainbow across the entire Procgen benchmark. Mean and standard deviation are calculated over 5 trials, each with a different seed.

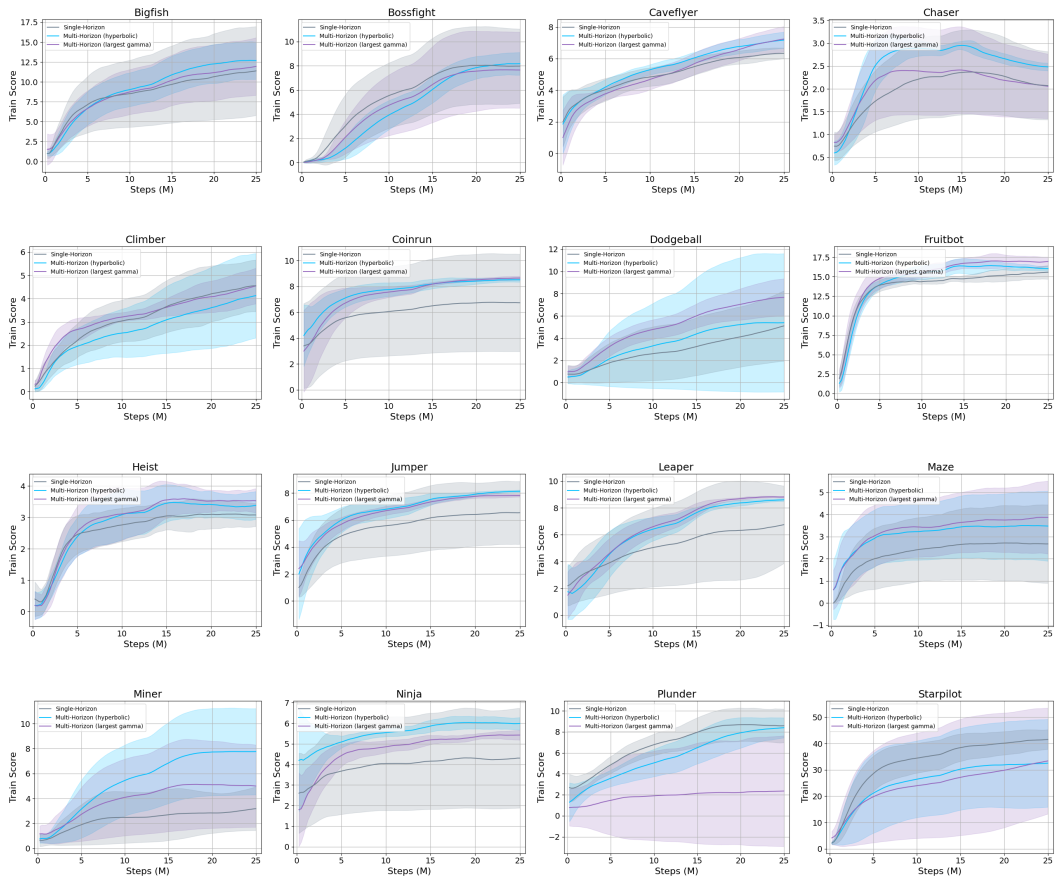


Figure 6: Train performance for Single-Horizon, Multi-Horizon (largest gamma) and Multi-Horizon (hyperbolic) variants of Rainbow across the entire Procgen benchmark. Mean and standard deviation are calculated over 5 trials, each with a different seed.

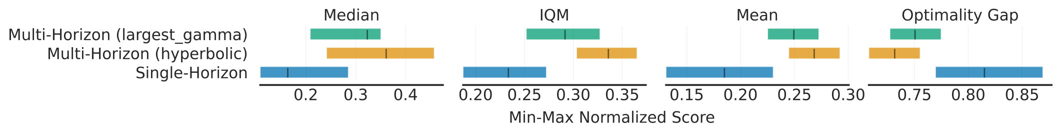
5.4 Train performance on all Procgen environments

Figure 6 shows the train performance of Single-Horizon, Multi-Horizon (largest gamma) and Multi-Horizon (hyperbolic discounting) Rainbow variants on all Procgen games. As expected, the agents training performance fares better than their test performance, as agents are trained only on a subset of 200 levels for each environment. The training performance analysis is similar to the test, with the exception of Heist which shows a consistent learning pattern for all three methods.

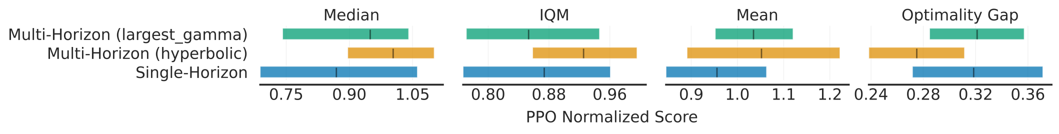
6 Evaluating statistical uncertainty

In order to reliably evaluate results and reduce statistical uncertainty in the comparisons, we analyze the performance of our approach as proposed in Agarwal et al. [2021]. This was all the more important since our experiments used only 5 seeds. In order to perform a robust evaluation of the results, we compute performance statistics across all games within the Procgen benchmark using the metrics as mentioned in the rliable paper. This also provides a holistic view of each method's performance across the benchmark, as each individual game characteristics and dynamics vary and hence the resulting performance for each method also varies. We briefly review the rliable metrics used in our results.

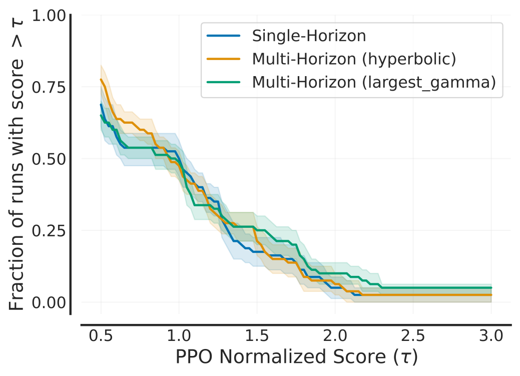
Interquartile Mean (IQM) takes the second and third quartiles (the middle 50%) of the runs combined across seeds and environments, and calculates the mean score. IQM is not affected by outliers and reduces the statistical uncertainty even with few runs. **Optimality gap (OG)** measures the



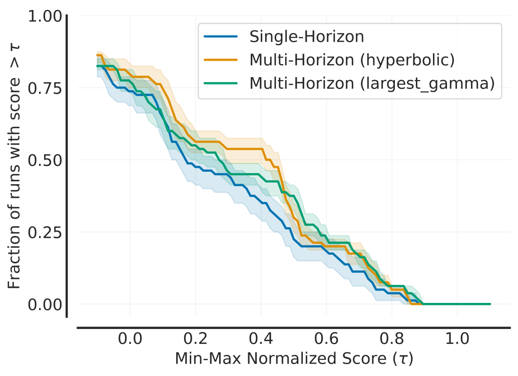
(a) Aggregate Metrics (Min-Max normalized)



(b) Aggregate Metrics (PPO normalized)



(c) Performance Profiles



(d) Performance Profiles

Figure 7: Performance comparison of Single-Horizon, Multi-Horizon (largest gamma), and Multi-Horizon (hyperbolic discounting) over all 16 Procgen environments. **(a)** Aggregate Metrics (Min-Max Normalized) including Mean, Mean, IQM and OG metrics for each method. **(b)** Aggregate Metrics (PPO Normalized) including Mean, Mean, IQM and OG metrics for each method. **(c)** PPO Normalized Performance Profile of the three methods. **(d)** Min-Max Normalized Performance Profile of the three methods.

proportion of performance that fails to meet a minimum threshold score of $\alpha = 1.0$, such that scores beyond α are not considered very important. For both IQM and OG, instead of taking point estimates, interval estimates using stratified bootstrap confidence intervals are computed. **Performance profile** of an algorithm depicts its score distribution as the fraction of runs above a certain normalized score. These profiles visualize the empirical tail distribution function of a random score, but with stratified bootstrap sampling to produce point-wise confidence bands. The higher the curve of a method, the better it is. The scores are normalized either using PPO mean (taken from results published as part of the *rliable* library [Agarwal, 2021]), or through the achievable min-max scores given for each environment (Appendix C, Normalization Constants of Cobbe et al. [2020]).

The results are shown in Figure 7 which confirm our earlier findings of Multi-Horizon (hyperbolic) variant performing better then Multi-Horizon (largest gamma), alluding to the fact that for procedurally generated environments and as such any environment where we want an agent to learn a robust policy (unlike memorizing trajectories), learning over multiple horizons can not be just thought of as an auxiliary task. By discounting hyperbolically and *utilizing value estimates from all time horizons to make decisions (acting policy)* is an important consideration for learning intelligent behavior.

7 Conclusion

We report initial results on the use of multi-horizon learning using advantage-based action selection in procedurally generated environments. Our results indicate that agents which model value estimates over multiple timescales generally perform better than their single-horizon counterparts. Moreover, advanced discounting functions like hyperbolic that are leveraged through the use of multiple timescales perform better or at par with exponentially-discounted, multi-horizon (largest gamma) variants. Agents which learn over multiple horizons but act myopically (smallest gamma) can also sometimes fare better than single horizon baselines, by virtue of their ability to learn from farther horizons. The reason why no particular method performs well across all environments can be attributed to the fact that an agent's prior belief of the risk in the environment influences the specific discounting function used [Fedus et al., 2019]. However, for trials across the full suite of 16 environments over 25M timesteps, the performance profile of multi-horizon (hyperbolic) is higher (and better) than the single-horizon and multi-horizon (largest gamma) variants. Our contribution can be seen as validating the impact of multi-horizon learning on an agent's policy through learning accurate value estimates, achieving higher sample efficiency and better generalization.

References

Dimitri P Bertsekas and John N Tsitsiklis. Neuro-dynamic programming: an overview. In *Proceedings of 34th IEEE conference on decision and control*. IEEE, 1995.

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.

William Fedus, Carles Gelada, Yoshua Bengio, Marc G Bellemare, and Hugo Larochelle. Hyperbolic discounting and learning over multiple horizons. *arXiv preprint arXiv:1902.06865*, 2019.

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Hasselt, Marc Lanctot, and Nando Freitas. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*. PMLR, 2016.

Nasik Muhammad Nafi, Raja Farrukh Ali, and William Hsu. Hyperbolically discounted advantage estimation for generalization in reinforcement learning. In *Workshop on Decision Awareness in Reinforcement Learning, ICML*, 2022.

Dominik Schmidt and Thomas Schmied. Fast and data-efficient training of rainbow: an experimental study on atari. *Deep RL Workshop NeurIPS*, 2021. URL `https://arxiv.org/abs/2111.10247`.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018.

Karl Cobbe, Chris Hesse, Jacob Hilton, and John Schulman. Leveraging procedural generation to benchmark reinforcement learning. In *ICML*, pages 2048–2056. PMLR, 2020.

Richard S Sutton. Td models: Modeling the world at a mixture of time scales. In *Machine Learning Proceedings 1995*, pages 531–539. Elsevier, 1995.

Saori C Tanaka, Kenji Doya, Go Okada, Kazutaka Ueda, Yasumasa Okamoto, and Shigeto Yamawaki. Prediction of immediate and future rewards differentially recruits cortico-basal ganglia loops. In *Behavioral economics of preferences, choices, and happiness*, pages 593–616. Springer, 2016.

James E Mazur. Choice, delay, probability, and conditioned reinforcement. *Animal Learning & Behavior*, 25(2):131–147, 1997.

Leonard Green, Nathanael Fristoe, and Joel Myerson. Temporal discounting and preference reversals in choice between delayed outcomes. *Psychonomic Bulletin & Review*, 1(3):383–389, 1994.

Zeb Kurth-Nelson and A David Redish. Temporal-difference reinforcement learning with distributed representations. *PLoS One*, 4(10):e7362, 2009.

Zhongwen Xu, Hado P van Hasselt, and David Silver. Meta-gradient reinforcement learning. *Advances in neural information processing systems*, 31, 2018.

Craig Sherstan, Shibhansh Dohare, James MacGlashan, Johannes Günther, and Patrick M Pilarski. Gamma-nets: Generalizing value estimation over timescale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5717–5725, 2020.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International Conference on Machine Learning*, pages 1407–1416. PMLR, 2018.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In *AAAI conference on artificial intelligence*, 2018.

Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

Meire Fortunato, Mohammad Gheshlaghi Azar, Bilal Piot, Jacob Menick, Ian Osband, Alex Graves, Vlad Mnih, Remi Munos, Demis Hassabis, Olivier Pietquin, et al. Noisy networks for exploration. *Proceedings of the International Conference on Representation Learning (ICLR)*, 2018.

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.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In *ICLR*, 2016.

Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3 (1):9–44, 1988.

Pablo Samuel Castro, Subhodeep Moitra, Carles Gelada, Saurabh Kumar, and Marc G. Bellemare. Dopamine: A Research Framework for Deep Reinforcement Learning. 2018. URL `http://arxiv.org/abs/1812.06110`.

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G Bellemare. Deep reinforcement learning at the edge of the statistical precipice. *NeurIPS*, 2021.

Rishab Agarwal. rliable. `https://github.com/google-research/rliable`, 2021.

Supplementary Materials

A Setup

The software used included Python (3.8.x) and PyTorch (1.12), with CUDA (11.4). Our computing infrastructure involved using 2 GPU servers, each having an Intel Xeon 4215 processor (3.50GHz), 16 physical cores, 1 TB RAM, and 8 Nvidia A40 GPUs. Each experiment was run using one GPU, and took between around 1 day (about 25 hours), totaling 4.5 days (about 108 hours) for the set of experiments (16 envs x 3 seeds x 2 methods + 4 envs x 3 seeds x 1 method). The experiments did not use vectorized environments.

B Hyperparameters

We provide the hyperparameters used in our implementation in Table 1.

hyperparameters	**values**
Q-target update frequency	8,000 steps
Importance sampling beta for PER	0.4
n-step	3
σ_0 for noisy linear layers	0.5
Learning rate	0.00025
batch size	256
Adam e parameter	0.00002 (0.005/batch size)
parallel environments	64
Replay buffer size	$\approx$1M (2^{20})
Training starts at	20,000 steps
γ (single-horizon)	0.99
number of γ	5
γ_{max}	0.99
k (hyperbolic exponent)	0.1
integral estimate	lower
$\gamma_{interval}$ values	[0.374, 0.608, 0.755, 0.847, 0.904]
γ values	[0.906, 0.951, 0.972, 0.985, 0.99]
number of seeds per environment	5
random seed values	[64331, 74330, 95762, 2822995, 801604]
Procgen distribution mode	Easy
Procgen num levels (training)	200
Procgen num levels (evaluation)	0
Procgen start level	0

Table 1: Hyperparameters for Rainbow

for gamma$_max$: $0.99, hyp_exp : 0.1, num_gammas : 5 EvalGammas[0.3746720177677334, 0.608964914637322, 0.7554748190881345, 0.8470915620154031, 0.9043820750088$