Reinforcement Learning Sustainability Benchmark

Luca Strefezza

June 29, 2024

Contents

1 Context

This project addresses the energy consumption of deep reinforcement learning (DRL) solutions and their impact on the environment and business costs.

Beginning with the resurgence of the field following the development of *Deep Q-Networks* (DQN) by DeepMind in the early 2010s [1], there have been a number of algorithm proposals over time that with minor modifications to DQN or using a completely different paradigm (such as policy gradient methods) sought to improve the performance achieved by the learning agent.

Although the performance of the various solutions has been extensively studied and tracked, little effort has been directed toward understanding how the tweaks to the DQN introduced to improve performance impacted energy consumption, or what the cost of the alternative approaches developed was, per se and in comparison with previous solutions.

The motivation behind this project is to fill this gap by evaluating the trade-offs between performance and energy consumption for several widely used deep reinforcement learning (DRL) algorithms. Understanding these trade-offs is crucial for businesses and researchers who aim to optimize both performance and sustainability in their applications. This project aims to provide valuable insights into the energy efficiency of different DRL approaches, enabling informed decisions about their use in various contexts.

To reach this goal we train various reinforcement learning algorithms on the same task, the choice of which is discussed in section 3.2 on page 5. Section 3.1 on the next page describes the selected algorithms, whose choice was made taking into account that DRL algorithms can be devided in two main categories: *value based* (i.e. algorithms based on the approximation of a value function, be it the state-value function or the action-value function) and *policy gradient*. The latter are methods that approximate directly the policy, and includes as a special case the *actor-critic methods*, which approximate simultaneously a policy (said actor) and a value function (said critic).

2 Goals

The primary goal of this project is to benchmark the energy consumption and performance of various deep reinforcement learning algorithms. Specifically, we aim to:

1. evaluate the energy consumption of different DRL algorithms when trained on the same task;
2. compare the performance of these algorithms in terms of their ability to achieve high scores on the given task;
3. analyze the trade-offs between performance and energy consumption to identify the most efficient algorithms;
4. provide a comprehensive report that can guide practitioners in selecting the appropriate DRL algorithms based on specific use-case requirements.

By achieving these goals, the project will contribute to the broader understanding of the sustainability implications of deep reinforcement learning technologies.

3 Methodological Steps to Conduct to Address the Goals

The methodology used follows from the basic idea of this benchmark: to execute all the algorithms for the same number of environment interactions, so that we can compare the score they achieve and the energy consumption of each one of them. Additionally, a good comparison would be to take the score obtained by the lowest performer in this initial trial and re-train all the algorithms until they reach that score. This would allow us to compare how much time and energy each algorithm requires to achieve the same performance level. Unfortunately, time and resources constraints make retraining all algorithms unfeasible, so we will approximate this second comparison by using the returns from the logging of the training during the first trial. This logging includes the `global_step`, indicating the environment interaction we are at, and the `episodic_return`, which is the return of the episode (i.e., the score on which to compare), as well as all performance and power consumption data up to that point. By analyzing these logs, we will estimate how much time and energy each algorithm would take to reach the score obtained by the lowest performer in the initial trial.

The following sections outline the several key steps involved in the methodology adopted for this project.

3.1 Algorithms Selection

As stated, in our benchmark we will consider both value-based methods and policy gradient methods. The selected algorithms are chosen to represent a wide range of approaches within both categories.

3.1.1 Value-Based Methods

Value-based methods are algorithms based on the approximation of a value function. The algorithms we will test in this category are:

- **Deep Q-Network (DQN):** the first example of success in deep reinforcement learning, will serve as a sort of baseline for our benchmark.
- **RAINBOW [2]:** an advanced method that combines several improvements to the original DQN, that will also be tested individually to assess their individual contributions to energy consumption and performance. These are listed hereafter:
  - Double Q-Learning (Double DQN) [3];
  - Prioritized Experience Replay [4];
  - Dueling Network Architectures [5];
  - Multi-step Learning [6];
  - Distributional RL [7];
  - Noisy Nets [8];
- **Self-Predictive Representations (SPR) [9]:** a more advanced method introduced in recent research, which leverages self-predictive representations to improve efficiency.

3.1.2 Policy Gradient Methods

Policy gradient methods approximate the policy directly and include as a special case the actor-critic methods, which simultaneously approximate a policy and a value function. The algorithms we will test in this category are:

- **REINFORCE [10, Chapter 13]:** a basic policy gradient method, or its variant REINFORCE with baseline (also known as Vanilla Policy Gradient, VPG).
- **Proximal Policy Optimization (PPO) [11]:** a popular and efficient policy gradient method that uses a clipped objective to improve training stability.
- **Deep Deterministic Policy Gradient (DDPG) [12]:** an algorithm that combines policy gradients with deterministic policy updates for continuous action spaces.
- **Twin Delayed DDPG (TD3) [13]:** an improvement over DDPG that addresses function approximation errors through various techniques, such as delayed policy updates and target policy smoothing.
- **Data-Regularized Q (DRQ) [14]:** a method that incorporates data augmentation to regularize the training of Q functions, improving performance and stability.

3.2 Task Selection

Regarding the task on which to compare the algorithms, there were several suitable candidates: Atari 100k, one of the continuous control task of the DeepMind Control Suite, or one of the many other task (besides Atari) included in OpenAI Gymnasium (formerly Gym), and so on. After various tests and research we opted for the Atari 100k, a discrete task that consists of playing one of the Atari games for 100 000 environment interactions.

The reason for this choice is multifaceted. Atari 100k is a widely used benchmark in the DRL community, the wealth of prior research and baseline results available facilitates a more straightforward validation and comparison of our experimental results with those from other studies and algorithms. It is well suited for evaluating the performance of almost all popular DRL algorithms, ensuring a comprehensive assessment. Additionally, Atari games provide a range of different challenges, including planning, reaction time, and strategy, making it a robust benchmark for assessing general DRL capabilities.

Moreover, the discrete nature of Atari 100k simplifies the implementation and comparison of algorithms, as continuous control tasks often require additional considerations and modifications. Finally, the 100 000 interactions limit strikes a balance between providing enough data for meaningful evaluation and being computationally feasible within our resource constraints, especially considering the large number of experiments required for each algorithm, as detailed below.

These factors combined make Atari 100k a practical and effective choice for our benchmark, enabling us to achieve our project goals efficiently.

3.3 Experiment Setup

In this section we will address all the decisions made in the setup of the experiments.

3.3.1 Number of Runs

In determining how many runs to carry out during the experimentation and testing of a reinforcement learning algorithm, at least two fundamental aspects must be taken into account: the high variance of reinforcement learning, and thus its high susceptibility to randomness, and the evaluation of the generality of the algorithm, which must therefore be tested in several different environments in order to actually prove that it is capable of solving multiple problems and not be ultra-specialized on a single use-case.

In addressing the first aspect we can refer to the literature to get an idea of how many runs with different seeds are usually used to alleviate this problem. If in the early days of RL (and not DRL) the number of runs stood at around 100 and in any case did not fall below 30 at least until the introduction of ALE (Arcade Learning Environment) [15] included, with the advent of DRL the number of runs was consistently reduced to 5 or less because of the high cost in terms of time and resources per run. Although this has been the standard for years, a more recent work [16] has shown that using point estimates of aggregate performance such as mean and median scores across tasks is not

the best way to summarize results because it ignores the statistical uncertainty inherent in using only a few runs.

In particular, the study points out that in the case of Atari at least 100 runs per environment are required to obtain robust results, a value that is, however, impractical in reality. In our case we will still be forced to limit ourselves to a maximum of 10 runs per environment, but it should be considered how this is a less significant problem in our case than in other studies, since we are not attempting to advance the state of the art performance of DRL algorithms, but have instead a focus on energy consumption, which should in any case remain constant regardless of the actual learning of the agent, which is instead related to randomness.

Despite this, we will still attempt to use, in addition to the more classic and popular metrics such as the point estimates mentioned above, other metrics suggested in [16], designed precisely to obtain more efficient and robust estimates and have small uncertainty even with a handful of runs, not being overly affected by outliers.

The implementation of the experiments will be based on the cleanrl project, which provides a set of high-quality, standardized RL implementations. We will tweak the algorithms already implemented in cleanrl to match our use case and add the missing ones, adhering to the same philosophy and implementation principles to maintain consistency and comparability.

1. **Environment Setup:** All algorithms will be implemented and run in a controlled environment to ensure consistent comparison.

2. **Training Procedure:** Each algorithm will be trained for the same number of environment interactions to allow for a fair comparison of energy consumption and performance.

3. **Energy Measurement:** The energy consumption of each algorithm will be measured using appropriate tools and methodologies.

4. **Performance Evaluation:** The performance of each algorithm will be evaluated based on the scores achieved in the Atari 100k benchmark.

5. **Multiple Runs:** Each algorithm will be run 10 times on 15 different games, using different random seeds for each run to ensure robustness of the results.

6. **Additional Benchmark:** As an additional comparison, all algorithms will be re-trained until they reach the score obtained by the lowest performer in the initial benchmark, and their energy consumption and time to achieve this score will be recorded. This will be approximated using the returns from the logging of the training during the first trial.

3.4 Data Collection and Analysis

1. Collect data on energy consumption and performance for each algorithm.

2. Analyze the data to identify trade-offs between performance and energy consumption.

3. Generate visualizations and statistical analyses to present the findings.

References

[1] V. Mnih *et al.*, *Playing atari with deep reinforcement learning*, 2013. arXiv: 1312.5602 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1312.5602.

[2] M. Hessel *et al.*, "Rainbow: Combining improvements in deep reinforcement learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, 2018.

[3] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," *arXiv preprint arXiv:1509.06461*, 2016.

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

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

[6] J. Peng and R. J. Williams, "Incremental multi-step q-learning," in *Machine Learning Proceedings 1994*, Elsevier, 1994, pp. 226–232.

[7] M. G. Bellemare, W. Dabney, and R. Munos, "A distributional perspective on reinforcement learning," *arXiv preprint arXiv:1707.06887*, 2017.

[8] M. Fortunato *et al.*, "Noisy networks for exploration," *arXiv preprint arXiv:1706.10295*, 2017.

[9] M. Schwarzer, A. Anand, R. Goel, R. D. Hjelm, A. Courville, and P. Bachman, "Data-efficient reinforcement learning with self-predictive representations," *arXiv preprint arXiv:2007.05929*, 2020.

[10] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[11] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[12] T. P. Lillicrap *et al.*, *Continuous control with deep reinforcement learning*, 2019. arXiv: 1509.02971 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1509.02971.

[13] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International Conference on Machine Learning*, PMLR, 2018, pp. 1587–1596.

[14] I. Kostrikov, D. Yarats, and R. Fergus, "Image augmentation is all you need: Regularizing deep reinforcement learning from pixels," *arXiv preprint arXiv:2004.13649*, 2020.

[15] 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, Jun. 2013, ISSN: 1076-9757. DOI: 10.1613/jair.3912. [Online]. Available: http://dx.doi.org/10.1613/jair.3912.

[16] R. Agarwal, M. Schwarzer, P. S. Castro, A. Courville, and M. G. Bellemare, *Deep reinforcement learning at the edge of the statistical precipice*, 2022. arXiv: 2108.13264 [cs.LG]. [Online]. Available: https://arxiv.org/abs/2108.13264.