An empirical study of the naïve REINFORCE algorithm for predictive maintenance of industrial machines

Rajesh Siraskar

June 15, 2023

Abstract

The 14th century friar William of Ockham has been attributed for giving us the Occam's razor principle – when there are two competing "theories" predicting the same phenomenon, one should prefer the *simpler* of two.

In this empirical study, we document the performance of a simple, early reinforcement learning algorithm known as REINFORCE, implemented for a predictive maintenance problem. We compare a very naïve implementation of REINFORCE against the predictions of industry-grade Stable-Baselines3 (SB-3) implementations of three advanced algorithms namely, Deep Q-Network (DQN), Advantage Actor Critic (A2C) and Proximal Policy Optimization (PPO). Our broad goal was to understand the performance under various scenarios such as simulation based environment, three sets of *real* tool-wear data, added levels of noise and a random chance of break-down. Model performance was measured by how accurately the predictive maintenance agent suggested tool replacement as compared to a deterministic preventive maintenance rule based on tool-wear threshold.

Our findings indicate that the REINFORCE performs significantly well, for this particular problem. Across variants of environment, the REINFORCE algorithm demonstrated an average F1 performance of 0.836 as against 0.383 for A2C, 0.471 for DQN and 0.402 for PPO. As a measure of stability, the overall standard deviations for REINFORCE was 0.041, while A2C, DQN and PPO standard deviations were 0.059, 0.029 and 0.070 respectively. Across precision on tool replacement, REINFORCE was better by 0.354 basis points than the best of advanced algorithms and demonstrated a variance lower by 0.004. While the REINFORCE demonstrated better performance for each variant, it was observed that the training was unstable occasionally producing poor performance models. On the other hand the SB-3 implementations training was more stable almost always producing models with an F1 in the range 0.47-0.50.

1 Introduction

Introduced in 1992, the REINFORCE algorithm is considered as a basic reinforcement learning algorithm. It is a policy-based, on-policy as well as off-policy algorithm, capable of handling both discrete and continuous observation and action domains.

In practice the REINFORCE algorithm is considered as "weak" learner and superseded by several algorithms developed since. Most notably the Q-Learning and its deep-neural network version, the DQN, followed by Actor-Critic and one of the most robust modern day algorithms, the PPO.

Reinforcement Learning in Robotics: A Survey - Jens Kober J. Andrew Bagnell Jan Peters - Initial gradient-based approaches such as finite differences gradients or REINFORCE (Williams, 1992) have been rather slow. The weight perturbation algorithm is related to REINFORCE but can deal with non-Gaussian distributions which significantly improves the signal to noise ratio of the gradient (Roberts et al., 2010). Recent natural policy gradient approaches (Peters and Schaal, 2008c,b) have allowed for faster convergence which may be advantageous for robotics as it reduces the learning time and required real-world interactions.

xxx

xxx

xxx

xxx

xxx

1.1 Algorithm timelines

- 1947: Monte Carlo Sampling
- 1959: Temporal Difference Learning
- 1989: Q-Learning

- 1992: REINFORCE
- 2013: DQN
- 2016: A3C
- 2017: PPO

2 The REINFORCE algorithm

Three key features of any RL algorithm:

1. Policy: π_θ = Probablities of all actions, given a state. Parameterized by θ
2. Objective function:
$$\max_\theta J(\pi_\theta) = \mathop{\mathbb{E}}_{\tau \sim \pi_\theta} [R(\tau)] \tag{1}$$
3. Method: Way to udate the parameters = Policy Gradient

2.1 Policy gradient numerical computation

1. Plain vanilla:
$$\nabla \theta J(\pi_\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \, [\, \sum_{t=0}^{T} R_t(\tau) \, \nabla_\theta \ln \pi_\theta(a_t|s_t) \,] \tag{2}$$
2. With Monte Carlo sampling and approximation: $\nabla_\theta J(\pi_\theta) \approx [\, \sum_{t=0}^{T} R_t(\tau) \, \nabla_\theta \ln \pi_\theta(a_t|s_t) \,]$
3. With baseline: $\nabla_\theta J(\theta) \approx [\, \sum_{t=0}^{T} (R_t(\tau) - b(s_t)) \, \nabla_\theta \ln \pi_\theta(a_t|s_t) \,]$
4. Where, baseline does not change per time-step, it is for the entire trajectory
5. One baseline option: V^π - leads to Actor-Critic algorithm
6. Simpler option: Average returns over trajectory: $b = \frac{1}{T} \sum_{t=0}^{T} R_t(\tau)$

Algorithm

3 About Stable-Baselines-3

- SB3- paper (Raffin et al., 2021), Raffin et al. (2021)
- sb-3 main doc – (SB3, b)
- sb-3 ppo doc – (SB3, a)

4 Method

We normalize the tool wear and other state features, $x \in [0,\ 1] \subset \mathbb{R}$. This allows for adding white noise of similar magnitudes across experiments of different data-sets

5 Results

6 Discussion

7 Conclusion

References

Stable-baselines3 docs - reliable reinforcement learning implementations, a. URL https://stable-baselines3.readthedocs.io/en/master/modules/ppo.html#how-to-replicate-the-results.

Ppo, b. URL https://stable-baselines3.readthedocs.io/en/master/index.html. Accessed: 2023-05-14.

Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *J. Mach. Learn. Res.*, 22(1), jan 2021. ISSN 1532-4435.