# Evaluate-Reinforcement-Learning-Algorithms-Correctly

A simple but proper framework to accurately evaluate RL algorithms and fair comparison between different RL algorithms

Introduction:

unlike considering diffrences Reinforcement Learning has with other paradigms of machine learning (e.g. supervised learning, semi-supervised learning , unsupervised learning,...) , RL algorithms should be tested and evaluated in a specific framework that hold all necessary circumstances needed. in this repo with different examples we will try to make it clear how RL algorithms shuld be

During my master after enrolling in some deep learning courses and get my hands dirty with hot topics in the field back then (2022) including Meta learning, Few-shot Learning, multi-task learning and …, I tried to go beyond causual learning procedures in Reinfrocement learning algorithms. I started to study general pourpose model-free RL algorithms. When I say general pourpose algorithms, I mean algorithms that are not fine tuned or tailored to efficiently work in specific environments. Take DDPG or Double DQN algorithms. Both are general algorithms that can learn any model-free environment (of course not any environment to be specific 😊 there are many cases we cant even solve them with RL or any thing else).

I studied state of the art algorithms in continous and discrete action spaces. The problem was Hunger Games😉. The algorithms need many iterations and interactions to learn an moderate policy if not an near optimal (may assume a good policy if not optimal one). And I tried to solve this problem (yes I was kind of crazy at that time)

I did it however and developed an stochastic off-policy algorithm called TDS which outperforms DDPG, TD3 and SAC. (the work Is under peer review and I will upload it on my Github as soon as it is printed)

Propblem Statement:

The problem was ….. let me provide an example first. Back then I wrote an new algorithms which was simple and It was ouperforming many state of the art algorithms in discrete action spaces(DDQN) if they compared according to the number of episodes.

But all of the time my code spite of being simple would take