Deep Reinforcement Learning Assignment

Daniel Koltai
SN: 2123303

Part 1: Generalization of Trained Policies to Different Environment Dynamics

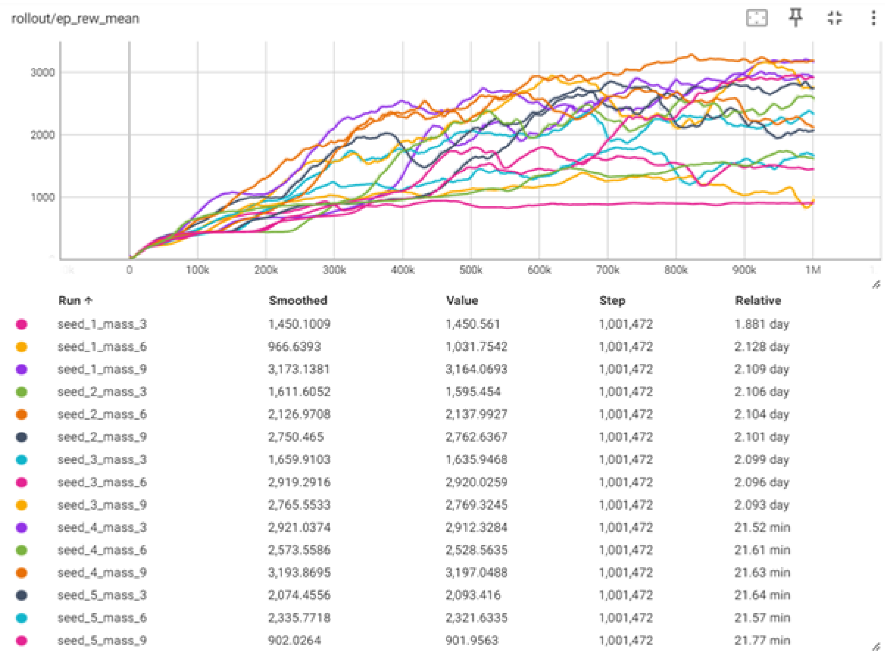
For the Hopper-v4 environment the Proximal Policy Optimization (PPO) model of stable-baselines3 was chosen for training the hopper agent. PPO was chosen for its capability to handle complex continous actions spaces and for its simplicity, stability and sample efficiency.

Hyperparameter tuning was performed manually by exploring several sets of hyperparameters. Annealing learning, clipping and exploration rates were tested with linear schedules. Tensorboard visualisations of trials with the different hyperparameters showed the highest stability for the default parameters of PPO. The benchmark parameters of PPO for the Hopper-v3 environment [1] were also tested and resulted in less stabile learning and significantly fewer total rewards.

Therefore, the default parameters of PPO (see below) were selected for training three different models with torso mass (3kg, 6kg, and 9kg) for 5 different random seeds for 1e6 timesteps.

```
(learning_rate=0.0003, n_steps=2048, batch_size=64, n_epochs=10,
   gamma=0.99, gae_lambda=0.95, clip_range=0.2, clip_range_vf=None,
    normalize_advantage=True, ent_coef=0.0, vf_coef=0.5,
   max_grad_norm=0.5, use_sde=False, sde_sample_freq=-1,
   rollout_buffer_class=None, rollout_buffer_kwargs=None, target_kl
   =None, stats_window_size=100, tensorboard_log=None,
   policy_kwargs=None, verbose=0, seed=None, device='auto',
   _init_setup_model=True)
```

Each model was then tested on a range of torso masses (3, 4, 5, 6, 7, 8, 9) kg. The test reproduced the generalization of trained policies of Rajeswaran et al. (2017) [2] for the Hopper-v4 environment. Each of the three model performs best with the torso masses close to its training torso mass thereafter showing a sharp decline in performance with increasing difference between training and testing torso mass.



	Run ↑	Smoothed	Value	Step	Relative
●	seed_1_mass_3	1,450.1009	1,450.561	1,001,472	1.881 day
●	seed_1_mass_6	966.6393	1,031.7542	1,001,472	2.128 day
●	seed_1_mass_9	3,173.1381	3,164.0693	1,001,472	2.109 day
●	seed_2_mass_3	1,611.6052	1,595.454	1,001,472	2.106 day
●	seed_2_mass_6	2,126.9708	2,137.9927	1,001,472	2.104 day
●	seed_2_mass_9	2,750.465	2,762.6367	1,001,472	2.101 day
●	seed_3_mass_3	1,659.9103	1,635.9468	1,001,472	2.099 day
●	seed_3_mass_6	2,919.2916	2,920.0259	1,001,472	2.096 day
●	seed_3_mass_9	2,765.5533	2,769.3245	1,001,472	2.093 day
●	seed_4_mass_3	2,921.0374	2,912.3284	1,001,472	21.52 min
●	seed_4_mass_6	2,573.5586	2,528.5635	1,001,472	21.61 min
●	seed_4_mass_9	3,193.8695	3,197.0488	1,001,472	21.63 min
●	seed_5_mass_3	2,074.4556	2,093.416	1,001,472	21.64 min
●	seed_5_mass_6	2,335.7718	2,321.6335	1,001,472	21.57 min
●	seed_5_mass_9	902.0264	901.9563	1,001,472	21.77 min

Figure 1: Rollout mean episode reward

Each model was then tested on a range of torso torso masses (3, 4, 5, 6, 7, 8, 9) kg. Figure 2 shows the results of the test. The test reproduced the generalization of trained polices of Rajeswarant et al. [2] for the Hopper-v4 environment. Each of the three model performs best with the torso masses close to its training torso mess whereafter showing a sharp decline in performance with increasing difference between training and testing torso mass. Rajeswaran et al had similar results in the Hopper-v3 environment with TRPO algorithms regards to the decline in performance as the difference of training and target torso masses grew [2].

Figure 1 presents the rollout mean episode reward for all trials with different torso masses and seeds. The mean episode reward flattens before 1 million timesteps, with some trials already being stabile while some showing some instability.

Standard deviation of the action probabilities during training steadily shrinked. Over time the policies become more confident in its action choice, hence more exploitative. Thus, showing stabile optimization.

Explained variance quickly (1e5 timesteps) approached 1 (exceeded 0.98) showing that the value function's predictions correlate well with the actual returns and it provided useful training signals to the policy.

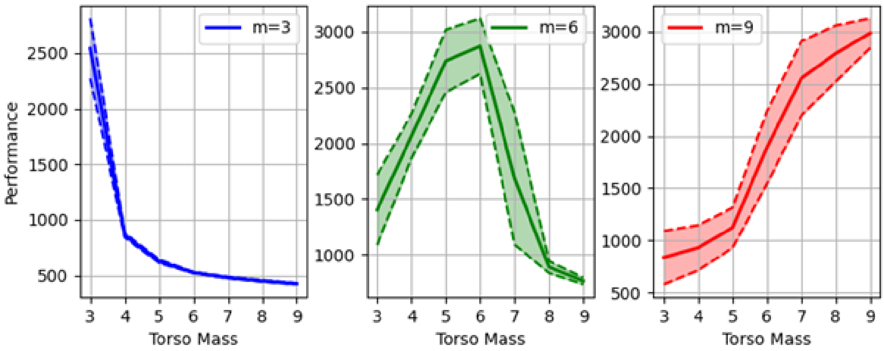


Figure 2: Performance of hopper policies when tested on target with different torso mass than training torso mass. The graphs with different colours show the testing results of models trained with three different torso masses. The policies use PPO with default parameters. The shaded regions show the 10th and 90th percentile of the return distribution.

Part 2: On-policy vs Off-policy Algorithms

On-policy Algorithm: Proximal Policy Optimization (PPO)

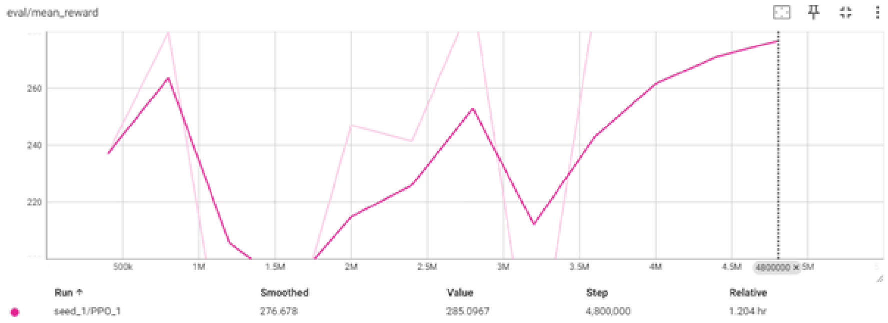
Proximal Policy Optimization (PPO) is chosen as the on-policy algorithm because of its strong performance across a wide range of environments and its stability due to gradient clipping, which prevents drastic policy updates. PPO is designed to address the inefficiencies of policy gradient methods and offers a good balance between sample complexity (i.e., number of samples needed to achieve a given performance) and computational complexity, making it suitable for environments like BipedalWalker-v3 where precise control is crucial.

PPO achieved a mean reward of approximately 285.63, with a standard deviation of 0.93. This is slightly below the target of 300, suggesting that while PPO is effective, there might be room for further tuning or it could be approaching its performance limit for this specific task.

Figure 3 shows the mean episode reward during evaluations throughout the training. The mean episode reward is slightly unstabile during training. It reaches 240 at timestep 5e5 and only exceeds 260 at timestep 4e6 and it decreases in its speed of increase towards timestep 5e6. The mean reward reaches a maximum value of 285 at the end of training. The maximum reward of this algorithm is thus close to 300, which counts as solid performance in the environment.

Other training metrics, such as explained variance, standard deviation of action policy distribution, entropy loss were showing the temporal patterns that indicate stabile learning.

Hyperparameter tuning was explored starting from SB3 RL-zoo benchmark's



Run ↑	Smoothed	Value	Step	Relative
● seed_1/PPO_1	276.678	285.0967	4,800,000	1.204 hr

Figure 3: Evaluation of mean episode reward

hyperparameters [3]. Anneling learning, exploration and clipping rates were tried with different ranges. Optuna was also used to find optimal hyperparameters for learning rate and exploration parameter. The original benchmark parameters performed the best eventually.

Off-policy Algorithm: Twin Delayed Deep Deterministic Policy Gradient (TD3)

TD3 is selected as the off-policy algorithm because it effectively addresses the overestimation bias found in other algorithms like DDPG by using a twin network architecture and delayed policy updates. These features make TD3 particularly effective for continuous action spaces, as it can more reliably converge to a stable and high-performing policy.

The mean reward obtained from TD3 is 302.94 with a standard deviation in reward of 45.9 This slightly surpassesses the target score of 300.

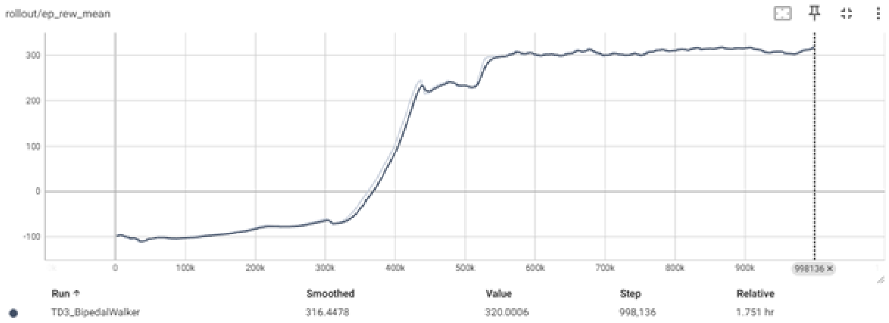
Hyperparameter tuning was explored starting from SB3 RL-zoo benchmark's hyperparameters [4]. The learning rate and exploration rate was adjusted. However, due to TD3 being computationally expensive hyperparemeter tuning was limited and the benchmark's hyperparameters yielded the highest performance.

Figure 4 shows the rollout mean episode reward. The figure indicates stabile learning, that plateaus from timestep 6e5.

Throughout training the actors's loss decreased to -20 and the critic's loss increased to 1.5.

Comparison of PPO and TD3

PPO is designed to ensure a more stabile learning by clipping the change in action distributions during updates to the policy. TD3 ensures the stability of learning by incorporating smoothing techniques, delayed policy updates and decreased



	Run ↑	Smoothed	Value	Step	Relative
●	TD3_BipedalWalker	316.4478	320.0006	998,136	1.751 hr

Figure 4: Rollout mean episode reward

overestimation bias in value estimates.

Multiple runs were performed with different seeds throughout hyperparameter tunings. In general, PPO had a small standard deviation and TD3 a higher standard deviation. However, there was a large variance between the results both in terms of the mean and standard deviation of the reward. PPO was always above 220 in rewards and TD3 always reached a reward of 300.

TD3 achieved a higher mean episode reward than PPO. TD3 just above 300 and PPO did not reach 300, but come close to it. TD3 had a relatively high standard deviation in mean episode reward at evaluation. In contrast PPO's standard deviation of mean episode reward was very low. This indicates that PPO performs more reliably with respect to the stochasticity of the environment.

PPO needed and order of magnitude more timesteps to achieve high rewards than TD3 (6e5 vs 5e6). However, PPO's wallclock time was 4 times faster than TD3. The computational cost of TD3 was thus a major disadvantage compared to PPO. TD3 is thus much more sample efficient than PPO, however each step is more computationally expensive in TD3.

These statements are based on the hyperparameters that were set to the algorithm during this research, for different hyperparameters they might not hold.

Overall, TD3 has high data efficiency and performs stabile learning while having a high computational cost and thus a longer wall clock-time. PPO is faster at computation, simpler to implent, effective at a broad range of environments with a good stability. PPO is less sample-efficient.

References

[1] `https://huggingface.co/sb3/ppo-Hopper-v3`

[2] A. Rajeswaran, S. Ghotra, B. Ravindran, and S. Levine, “Epopt: Learning robust neural network policies using model ensembles,” arXiv preprint arXiv:1610.01283, 2016.

[3] https://huggingface.co/sb3/ppo-BipedalWalker-v3

[4] https://huggingface.co/sb3/td3-BipedalWalker-v3