CMPUT628 - Deep Reinforcement Learning
Assignment 4

March 28, 2025

Dikshant
1877098

---

1. *Twin Delayed DDPG (TD3) implementation*

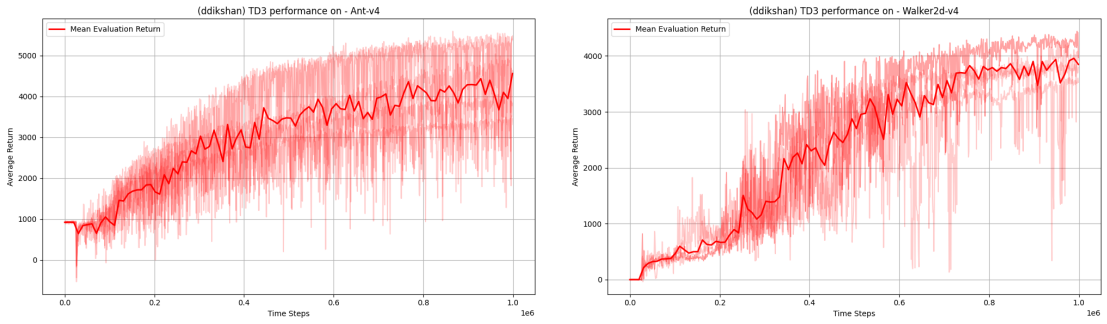
(a) TD3 is designed to address overestimation bias in continuous action spaces. The key improvements over standard DDPG include:

1. **Twin Critics**: Maintain two Q-networks to reduce overestimation
2. **Delayed Policy Updates**: Update policy less frequently than critics
3. **Target Policy Smoothing**: Add noise to target action predictions to stabilize training

The actor network takes the environment state as input, containing two fully connected hidden layers with 256 neurons each and a `tanh` activation output. The critic network processes state-action pairs through two fully connected hidden layers (256 neurons each), outputting Q-value estimates. To mitigate dependency on initial policy parameters, a purely exploratory policy is employed for the first 25,000 time steps. An off-policy exploration strategy adds Gaussian noise $\mathcal{N}(0, 0.1)$ to each action. Target policy smoothing is implemented by adding $\epsilon \sim \mathcal{N}(0, 0.2)$ to actions chosen by the target actor network, clipped to (-0.5, 0.5).

The algorithm implements delayed policy updates, modifying the actor and target critic networks every 2 iterations, with both target networks updated using a soft update parameter of $\tau = 0.005$. Each task is executed for 1 million time steps, with evaluations performed every 5,000 time steps to assess performance. During evaluations, the average return is calculated over 10 episodes and results are reported across 5 random seeds (Fig.1), with visualizations depicting the mean performance and individual run learning curves in lighter shade.

(b) Command to reproduce the results: `python train_td3.py --debug`



`Ant-v4` `Walked2d-v4`

Figure 1: TD3 agent learning curve on MuJoCo environments averaged over 5 seeds with lighter shade indicating individual runs

2. *Science*

I explored configurations that change the number of critics, update frequencies, policy noise levels, target update rates and types of exploration noise. The primary metrics that I tracked is average episodic reward calculated every few env steps.

1. *Twin critics vs single critic*: In traditional actor-critic methods, a single critic network can produce overly optimistic value estimates due to approximation errors. TD3 introduces two parallel critic networks that independently estimate state-action values. During value estimation, the algorithm uses the minimum value from these two critics. This approach helps reduce overestimation by providing a more conservative estimate of action values. Is the computational overhead truly justified by the performance gains? I suspect that the average return will be lower for single critic case.

2. *Actor-Critic update rates*: Delayed policy updates method deliberately updates the policy less frequently than the critic networks. This allows the value function to stabilize before modifying the policy's direction. By updating the actor network less often, the algorithm can develop a more accurate understanding of the environment before making significant policy changes. How sensitive is the algorithm to the timing of network updates? According to what's mentioned in the paper, if we update both actor and critic at the same rate, then there will be a lot more variance. Paper does not mention anything about delaying it even further. I don't know what to expect from the final plots as slowing the update rate can potentially help in reducing

the policy oscillations and make the value estimation more stable, however it will slow down the overall learning progress as well.

3. *Target Policy Smoothing*: Target policy smoothing adds another layer of complexity to the algorithm. The method introduces controlled noise to the target actor's action predictions. This noise is typically drawn from a small Gaussian distribution and then clipped to a narrow range. The goal is to prevent the policy from exploiting very specific, potentially misleading features of the value function. What happens if we increase this noise or we remove this fully? I expect that adding more noise in actions, can lead to more exploration and potentially learning more diverse strategies but it may also leads to less consistent policy learning and hence a lot of variance.

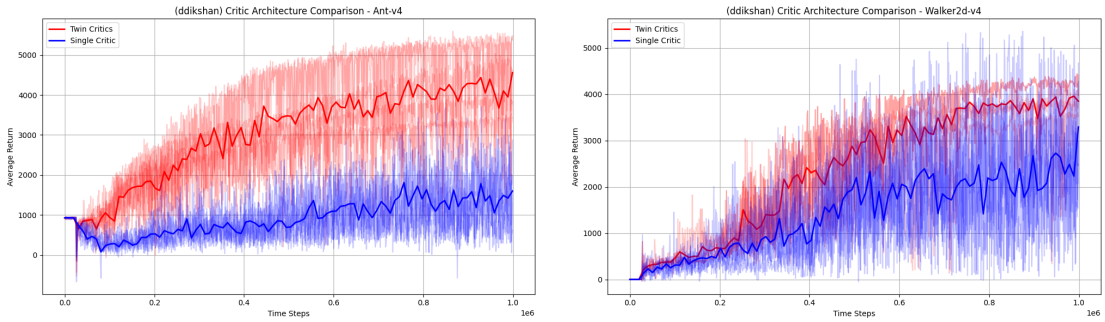
4. *Target Network update rates*: Target networks can be used to reduce the error over multiple updates. To ensure that the TD-error remains small, we update the target networks slowly ($\theta' \leftarrow \tau\theta + (1-\tau)\theta'$). How does the soft update parameter τ impact learning? I hypothesize that increasing τ can lead to higher variance in learning and rapid convergence of target networks and thus less stable learning. Lower value of τ can potentially provide more stable learning but it can be less responsive to recent changes.

5. *Exploration noise types*: Ornstein-Uhlenbeck(OU) noise provides temporally correlated noise which mimics momentum in action selection. TD3 authors state that noise drawn from OU process does not offer performance benefits without quantifying it. I wanted to explore what happens in case of OU noise in comparison to gaussian noise? According to what's mentioned in the paper, the performance in case of OU noise will be lower compared to gaussian noise.

I performed the following experiments to measure the effect of the above configurations. The first experiment serves as our baseline, maintaining the original TD3 paper's recommendations. Subsequent experiments isolate and modify specific components: removing one of the twin critics, altering the policy update frequency (from the default of 2 to 1 and 4 iterations), modifying policy noise levels (removing noise and increasing noise magnitude to 0.5), experimenting with different target network update rates (from slower 0.001 to faster 0.05 than default 0.005), and replacing Gaussian noise with OU noise. I focused on tracking average episodic rewards (same evaluation technique as stated above in part 1). Every experiment was ran over 5 seeds.

Ablation Results:

1. *Twin critics vs single critic*

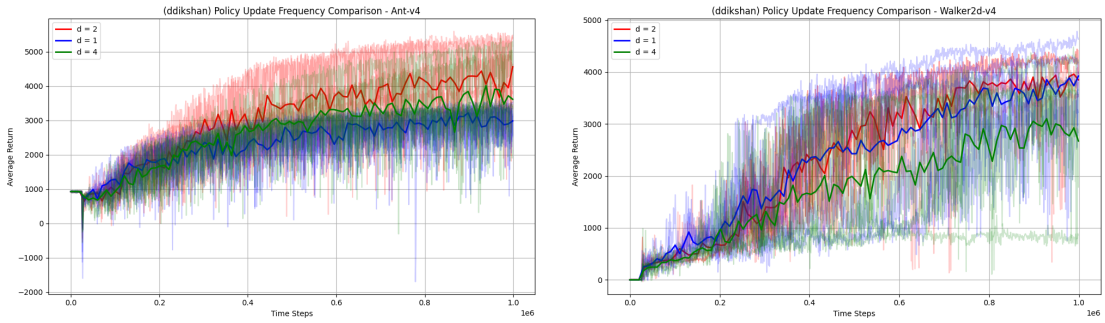


`Ant-v4` `Walked2d-v4`

Figure 2: Mean evaluation return for twin-critic vs single critic network averaged across 5 seeds with individual runs in lighter shade

Results clearly demonstrate the advantages of using twin critics over a single critic network. The single critic configuration produced lower return values in both environments, confirming that the computational overhead of maintaining two Q-networks is justified. This validates the core premise of TD3 - that using the minimum of two critics' estimates helps combat the tendency to overestimate action values which can lead to higher returns.

2. *Actor-Critic update rates*

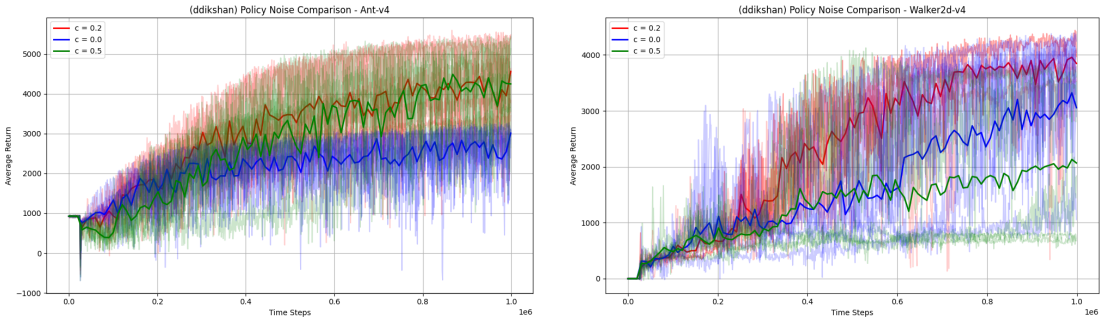


`Ant-v4` `Walked2d-v4`

Figure 3: Mean evaluation return for different policy update frequency d with default (2) averaged across 5 seeds with individual runs in lighter shade

Experiments examining different update frequencies between actor and critic networks show mixed results. The default delay (updating actor every 2 steps) doesn't clearly outperform updating both networks at the same rate in `Walker2d`. However, increasing the delay further (to 4 steps) seems to slow down learning progress. While the original TD3 paper suggested delays would reduce policy update variance, current results don't strongly support this claim. More experiments with additional random seeds and more environments would be needed to draw firmer conclusions about the benefits of delayed actor updates.

3. *Target Policy Smoothing*

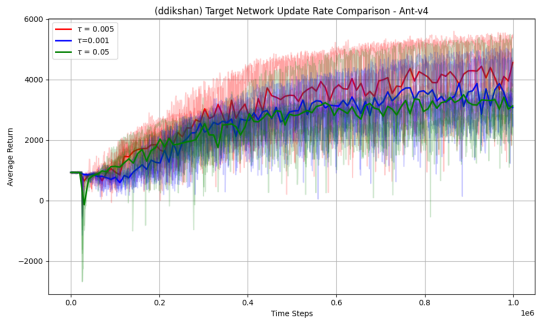


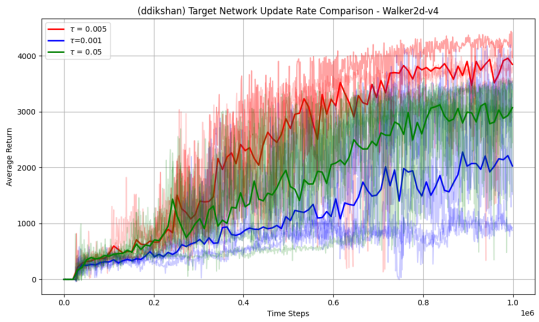
`Ant-v4` `Walked2d-v4`

Figure 4: Mean evaluation return for different policy noise c with default (0.2) averaged across 5 seeds with individual runs in lighter shade

Adding too much noise to target actions hurts performance, especially in the `Walker2d` environment where some agents failed to learn effectively. The `Ant` environment was more tolerant of increased noise. Removing policy noise entirely led to worse performance in both environments, suggesting some noise is needed for proper exploration and to prevent the algorithm from exploiting narrow peaks in the value function. The default noise level (0.2) appears to strike a good balance for both environments.

4. *Target Network update rates*



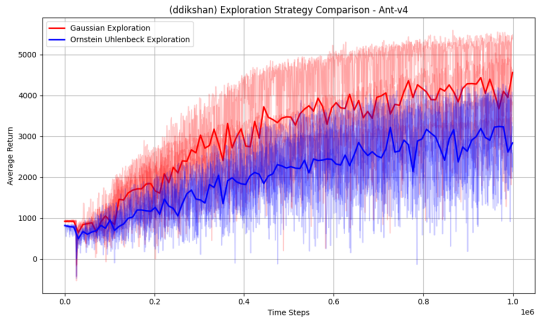


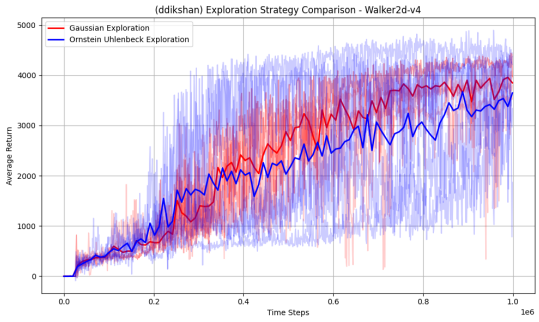
`Ant-v4` `Walked2d-v4`

Figure 5: Mean evaluation return for different soft update rate τ with default (0.005) averaged across 5 seeds with individual runs in lighter shade

A very small tau value (0.001) clearly slows down learning, as seen especially in `Walked2d`. While I expected a larger tau (0.5) would create unstable learning, the performance drop wasn't as dramatic as anticipated - returns were lower than the default (0.005) but variance wasn't substantially worse.

5. *Exploration noise types*





`Ant-v4` `Walked2d-v4`

Figure 6: Mean evaluation return for gaussian (default) vs OU exploration noise averaged across 5 seeds with individual runs in lighter shade

Results confirm the paper's finding that uncorrelated Gaussian noise performs better than OU noise. OU noise produced particularly unstable performance in the `Walked2d` environment, supporting the authors' decision to move away from the temporally-correlated noise that was used in the original DDPG algorithm.

Limitations:

There are few things which could have been improved for better analysis. I only used 5 random seeds instead of the planned 15, which means the results might not be truly representative of the algorithm's performance. The evaluation method was also different - I used an exploratory policy during evaluations, unlike the original paper which uses deterministic actions. This approach adds noise to the results and makes it harder to understand how the algorithm really performs. I didn't check the deterministic action part in the paper before so I have results for the stochastic actions. Ideally I should have performed a hyperparameter sweep for each new configuration but I found TD3 to be very slow with each seed taking about 12 hours to run. I submitted jobs for 15 seeds with hyperparameter sweep for all configurations but only managed to get results for the `Ant-v4` environment in last four days. I was also trying to capture the q-values to see how much overestimation actually happens in case of single critic. The results that I have shared in the report right now are only 5 seeds without deterministic actions while evaluation. The current results still provides some initial insights, it's more of a preliminary investigation than a definitive analysis of the TD3 algorithm.