RENDERING AND COMPARING DIFFERENT SAC VARIANTS

Jingyi Zhao

New York University Shanghai

1 INTRODUCTION

In this project, I experiment with different variants of REDQ Chen et al. (2021), and render the trained agents in two MuJoCo environments, Hopper and HalfCheetah. I will first compare the agents under different training epochs, and introduce a customized variant with $UTD = 5$ and reset. Then, I will show the rendered images of the simulated robots tested with different variants, and conduct a qualitative analysis.

In Section 2, I compare and discuss the performance of different variants of REDQ. In Section 3, I display the rendered images under different trained agents and conduct a qualitative analysis.

2 COMPARISON BETWEEN DIFFERENT VARIANTS

In this section, I compare the performance (i.e., test return) and computation efficiency of different SAC variants. I consider five different variants:

- Random agent, an untrained SAC agent taking random actions
- SAC agent trained for 100 epochs with $UTD = 1$, no action noise during rendering
- SAC agent trained for 300 epochs with $UTD = 1$, no action noise during rendering
- SAC agent trained for 300 epochs with $UTD = 1$, with action noise during rendering (here I perform ϵ-greedy for action selection with $\epsilon = 0.5$, namely, we choose deterministic optimal action and random action with equal probability)
- SAC agent trained for 300 epochs with $UTD = 5$ and network reset every 100 epochs, no action noise during rendering

2.1 STUDY ON TRAINING EPOCHS

Table 1 shows the performance (i.e., test return) of the five variants over an episode of 1000 steps.

Table 1: Test Return of Different Variants

Variant	**Hopper**	**HalfCheetah**
Random Agent	22	-287
SAC Epoch 100	1199	4251
SAC Epoch 300	1212	7284
SAC Epoch 300 with Noise	1200	6994
SAC Epoch 300 with UTD 5 + Reset	3333	8665

The random agent has very low test returns because random actions are generally unlikely to perform well. The test return increases as the number of epochs being trained increases. Nonetheless, the training efficiency is different under different environments. In the HalfCheetah environment, the variant with 300 training epochs performs much better than the one with only 100 training epochs. In comparison, we see only a small increase in performance of the variant with 300 training epochs with respect to the one with only 100 training epochs in the Hopper environment which is a more

challenging scenario. The increase in performance indicates that the agent needs more epochs to learn. The small scale of increase in performance in the Hopper environment may be because the learning phase has converged. Another possible solution is that the agent fails to learn better. Based on my experiment with the fifth variant, the latter reason seems more reliable.

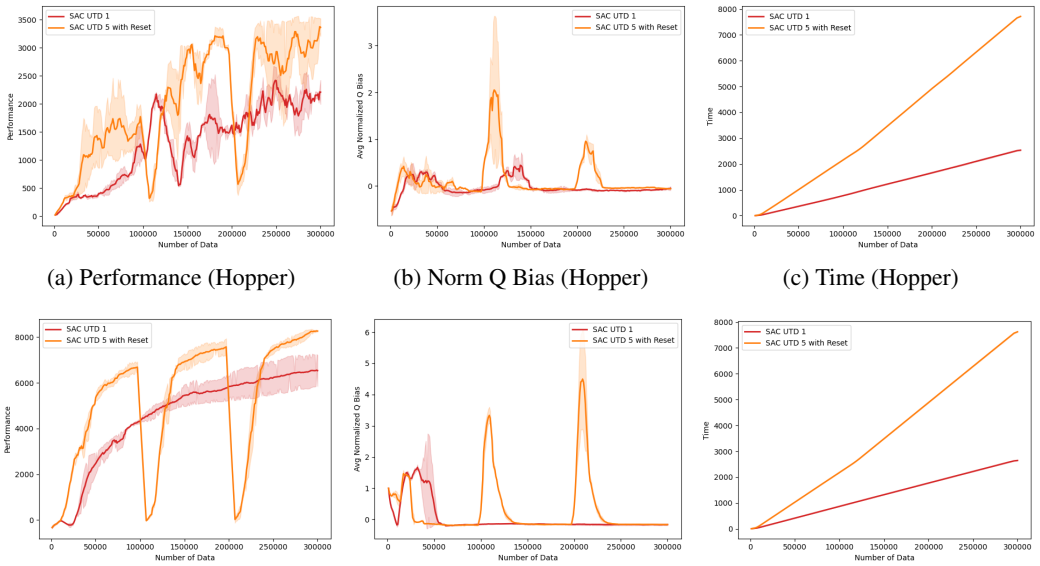
After introducing action noise during rendering, we see that the agent performs relatively worse than the one choosing deterministic optimal actions. The drop in return is obvious in the HalfCheetah environment but not in the Hopper environment. In fact, after several testing episodes in the Hopper environment, I find that the action noise sometimes results in a better performance. Since the agent trained for 300 epochs is not performing very well in the Hopper environment, it's likely that random explorations introduced by the action noise can actually provide better actions.

The fifth variant with a larger UTD ratio and reset gives the best results, a detailed analysis will be provided in Section 2.2.

2.2 Study on UTD and Reset

In part 1 project report, we've studied the influence of the UTD ratio and found that replacing $UTD = 1$ with $UTD = 5$ tends to improve the performance greatly. Also, Nikishin et al. (2022) also showed that combining frequent network reset and high UTD ratio can increase the performance by reducing primacy bias and encouraging sample efficiency. Hence, I'm inspired to propose my customized variant with $UTD = 5$ and frequent network reset. As is displayed in Table 1, this variant is able to give the best results among all the variants being experimented.

Figure 1: Experiment Results of UTD and Reset



(a) Performance (Hopper) (b) Norm Q Bias (Hopper) (c) Time (Hopper)

(d) Performance (HalfCheetah) (e) Norm Q Bias (HalfCheetah) (f) Time (HalfCheetah)

Figure 1 shows the experiment results of the SAC agent with $UTD = 1$ and the SAC agent with $UTD = 5$ and frequent reset (every 100 epochs). The latter variant performs much better than the former one and learns faster in the early stage. In terms of computation time, the latter version needs approximately four times the training time of the former variant. With a larger UTD ratio, we are conducting more network updates and encouraging sample efficiency, thus achieving better performances. At the same time, more network updates require larger computation time. Introducing reset reduces the risk of overfitting to the early data which may result from a large UTD ratio.

3 Rendering and Discussion

In order to visualize how the agents perform, we test the trained agents to guide a simulated robot to run in the MuJoCo environments. Specifically, we do the rendering in the Hopper and HalfCheetah environments for 1000 steps and save the rendered images. In this section, we display some of the rendered images of three variants: the random agent, the SAC agent trained with 100 epochs, and the SAC agent trained with 300 epochs with UTD ratio 5 and reset. The SAC agent trained with 100 epochs and the ones trained with 300 epochs rendered with and without action noises do not show much difference in the rendered images, hence we choose the above three variants for clearer illustration purposes.

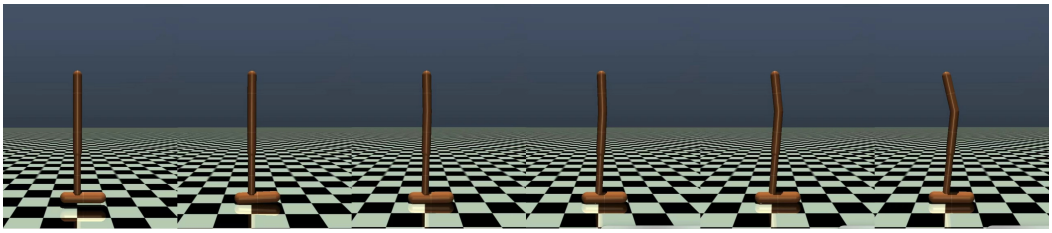
I will display and discuss the rendered images of the three variants in the Hopper environment in Section 3.1 and in the HalfCheetah environment in Section 3.2.

3.1 Rendering in the Hopper Environment

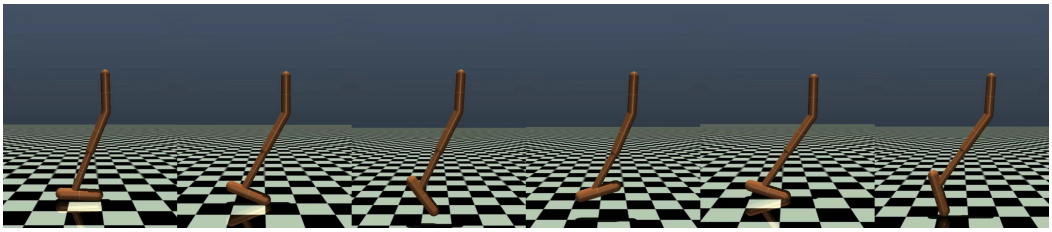
The random agent barely moves in the Hopper environment. In fact, though we conduct rendering for 1000 steps, we can only get approximately 40 rendered images for the random agent. The reason is that the random agent simply fails to learn how to walk and it gets stuck at the original position.

Both the SAC agent with 100 training epochs and the other one learns to walk, while the former fails down at approximately 400 steps and the later one manages to walk through the 1000 steps. As we can see from (b) and (c) from Figure 2, the former agent only keeps the top part stable and moves all the below parts, while the latter agent is able to keep the top two parts stable and only walks with the two parts below. In another word, the latter agent is better at keeping the majority of its body stable, hence it manages to walk pretty well.

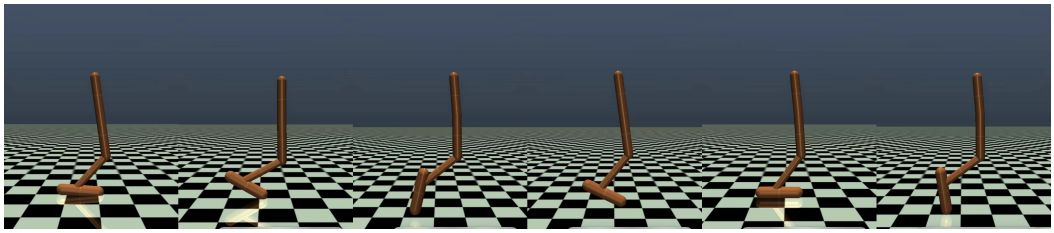
Figure 2: Rendered Images of Different Variants in Hopper Environment



(a) Random Agent



(b) SAC with Training Epoch 100



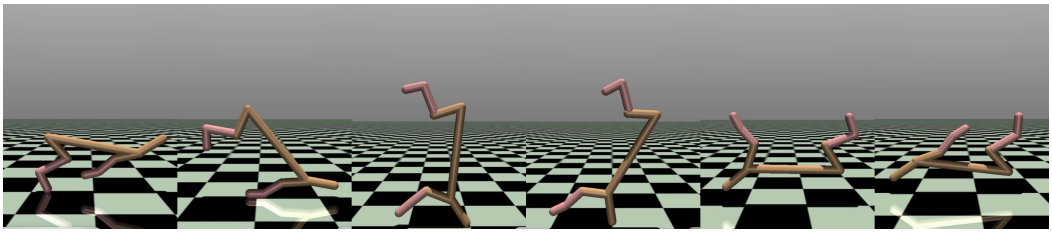
(c) SAC with Training Epoch 300, UTD 5 and Reset

3.2 Rendering in the HalfCheetah Environment

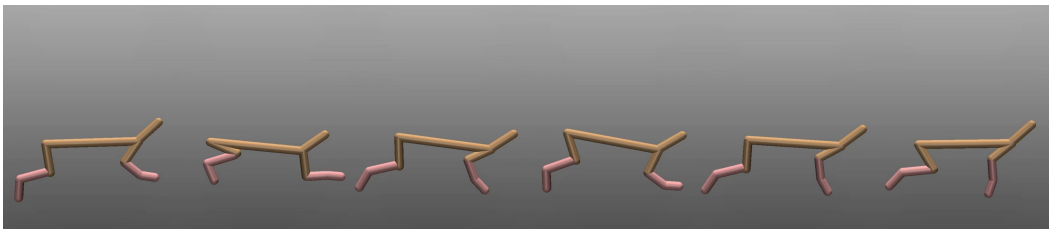
The random agent can barely move forwards. When it manages to move some step ahead, it quickly falls down. Then, it struggles to stand up again but fails. Since the random agent makes random actions, it will only move a little bit forward when its random actions happen to work. Nonetheless, this success won't last for more than a short time since it's easy to make random actions that will cause the agent to fall down. However, when it falls down, there is no way that it can stand up again.

Both the other two agents manage to run pretty well. Nonetheless, my last variant still performs better than the SAC agent trained with 100 epochs. Specifically, this variant runs faster than the other. Under closer observations, I find that this variant is better at balancing its foreleg and hind leg and maintaining the relative stability of its main body. In addition, it makes larger steps by stretching its legs more and hence runs faster.

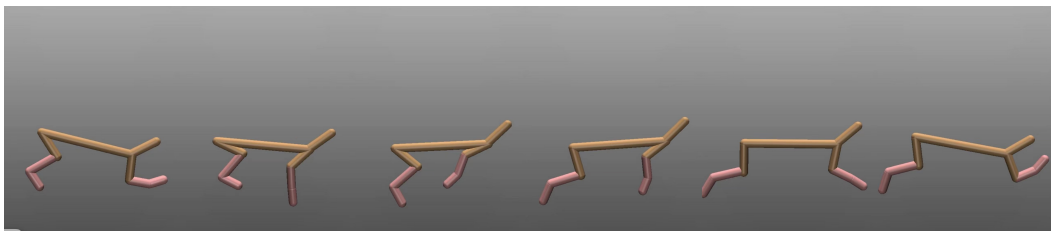
Figure 3: Rendered Images of Different Variants in HalfCheetah Environment



(a) Random Agent



(b) SAC with Training Epoch 100



(c) SAC with Training Epoch 300, UTD 5 and Reset

4 Conclusion

In this report, I present five variants of the SAC agent, compare their performances, display and discuss their rendering in two MuJoCo environments, Hopper and HalfCheetah. In general, the performance increases as the training epochs increase. Introducing action noises during rendering harms the performance in an easy scenario like the HalfCheetah environment, while in a trickier case like the Hopper environment, it may or may not harm the performance in that encouraging exploration is possible to result in better actions than the trained agent. The normal variant struggles to learn better in the Hopper environment as training epochs increase. In addition, introducing a higher UTD ratio and reset gives the best performance.

REFERENCES

Xinyue Chen, Che Wang, Zijian Zhou, and Keith Ross. Randomized ensembled double q-learning: Learning fast without a model, 2021. URL https://arxiv.org/abs/2101.05982.

Evgenii Nikishin, Max Schwarzer, Pierluca D'Oro, Pierre-Luc Bacon, and Aaron Courville. The primacy bias in deep reinforcement learning, 2022. URL https://arxiv.org/abs/2205.07802.