COL778: Principles of Autonomous Systems
Assignment 3

Harshit Goyal (2021MT10143)

April 2025

Please find all `video rollouts` at here.

1 DAGGER Implementation

1. For collecting rollouts during training, we allow the simulation for the maximum number of timesteps allowed by the environment (or till failure).
2. Trajectories are collected will be have atleast a target number of timesteps (total), which are added to the buffer.
3. At every timestep, give the current state of the agent (observation), the next action is chosen, with β probability using the expert's policy, otherwise, the agent's policy.
4. Batch size number of observations are sampled from the buffer. The MLP predicts the actions on these, which are training against the expert's predictions on these observations using $\mathcal{L}_2$ loss.

2 Environment Description

Environment	**Observation space dimension**	**Action space dimension**	Action type
`Hopper-v4`	11	3	`continuous`
`Ant-v4`	27	8	`continuous`

Table 1: Environment Description

3 Best Model Evaluation Metrics

3.1 `Hopper-v4` Agent

Agent	`n_layers`	`hidden_size`	**Episode mean length**	**Episode mean return**	**Episode stdev return**
`Expert`	-	-	884 – 888	2667 – 2676	467 – 482
`Imitation`	3	64	851.7	2646.5	631.6
`Imitation*`	3	128	945.8	2838.3	377.0
`Imitation`	3	256	953.0	2873.9	351.1

Table 2: `Hopper-v4` agent. Episode maximum length (set by environment) is 1000, training rollouts sampled till $15,000$ timesteps collected, 30 minutes of training on CPU, $\beta = 0.1$, replay buffer size $50,000$, batch size 512, `Adam(lr = 1e-3)` optimizer, model saving metric described in Metric for model saving. Evaluation over 1000 trajectories.

3.2 `Ant-v4` Agent

Agent	n_layers	hidden_size	Episode mean length	Episode mean return	Episode stdev return
`Expert`	-	-	999 − 1000	1009 − 1027	260 − 285
`Imitation`	4	128	1000.0	969.8	2.7
`Imitation`	4	256	1000.0	938.9	5.0
`Imitation*`	5	128	1000.0	977.0	2.9
`Imitation`	5	256	1000.0	943.8	2.1

Table 3: `Ant-v4` agent. Episode maximum length (set by environment) is 1000, training rollouts sampled till 15,000 timesteps collected, 60 minutes of training on CPU, $\beta = 0.1$, replay buffer size 50,000, batch size 512, `Adam(lr = 1e-3)` optimizer, model saving metric described in Metric for model saving. Evaluation over 1000 trajectories.

4 Metric for model saving

Environment	Reward
`Hopper-v4`	`healthy_reward + forward_reward - ctrl_cost`
`Ant-v4`	`healthy_reward + forward_reward - ctrl_cost - contact_cost`

Table 4: Reward functions for different environments

From Table 4 we see that the reward is a proxy for the episode length in both cases (`forward_reward` component). Hence we track the evaluation return, instead of episode length. Since evaluation is done on only around 15 trajectories, we use the metric

```
metric = eval_mean_return - eval_stdev_return
```

This intends to maximize the "worst performance" of the agent. To make this requirement even harder, the coefficient of `eval_stdev_return` can be increased, for example `metric = eval_mean_return - 3 * eval_stdev_return`.

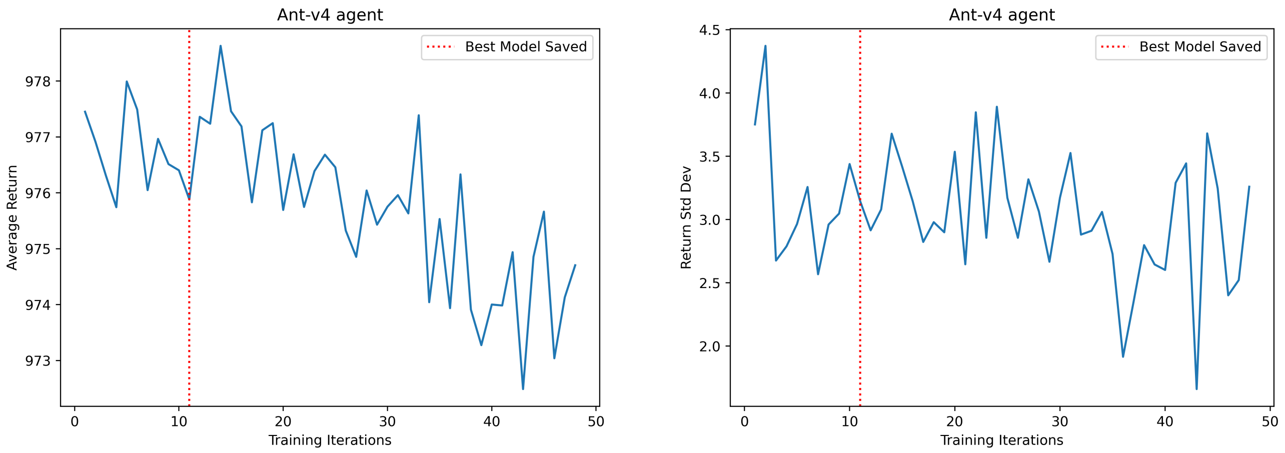
5 Observations

1. As evident in the training plots below (`Ant-v4`), the model doesn't get trained well and the return keeps reducing. Since the `Ant-v4` agent is 4-legged, it doesn't fall and hence the episode length is usually 1000 (full episode).

2. Even without training the policy network (`n_layers = 2, hidden_size = 64`), the average episode length is 1000 and for return, mean 957 with standard deviation 12.

3. We suspect this is because the return is mostly dominated by `forward_reward`, which always reaches it's full value, and there's not enough room left to learn skill.

4. We tried experimenting with reducing learning rate, Figure 1 but the return just **reduces** more slowly in this case. Similarly, changing `min_timesteps_per_batch` too didn't help.



Average Return

Standard Deviation of Return

Figure 1: Evaluation metrics of `Ant-v4` agent with `lr = 1e-4`

6 Training Plots

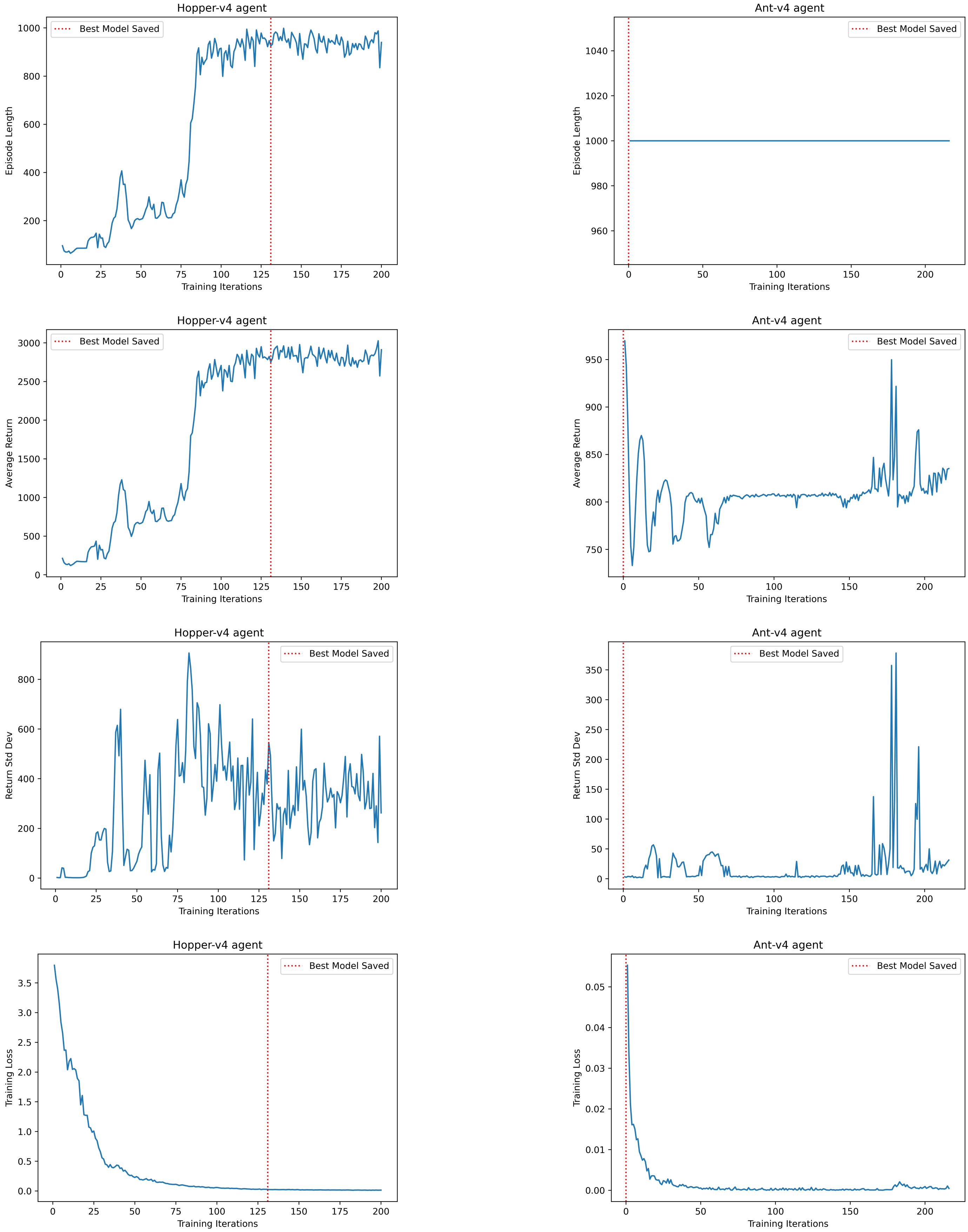


Figure 2: Evaluation metric while training. Mean Episode Length, Mean Return, Standard Deviation of Return, Training Loss (top to bottom). `Hopper-v4` (left), `Ant-v4` (right). Red dotted line indicates best-model.