Energy Management System using RL

Team Members:

1. Manish Reddy K (IMT2020019)
2. Harshadeep (IMT2020085)
3. Akhil Tavva (IMT2020124)

Objectives

- The primary objective is to optimize energy usage within a house. This includes managing heating and cooling systems to maintain comfortable temperatures while minimizing energy consumption and making use of solar energy generated efficiently.
- Another objective is that the environment models the dynamics of people within the house using airflow.
- The agent's goal is to maintain temperatures within these ranges to ensure comfort.

Overall, the goal of environment is to provide a platform for developing RL agents that can manage energy usage in a smart home.

Environment

Room Environment

- This environment simulates a house with controllable temperature and airflow systems. The observation space consists of the current temperature and energy level, both normalized within specific ranges.
- Rewards are computed based on maintaining temperature close to an ideal value and ensuring the energy level stays above a threshold, promoting efficient energy usage.
- The environment incorporates exploration through ε-greedy exploration, balancing exploration and exploitation.
- Upon reset, the environment randomly initializes the number of people in the room and may trigger home cleaning events, affecting energy levels.

House Environment

- The environment simulates a house with multiple rooms, including bedrooms and a living room. We initialize parameters such as initial temp, no of bedrooms etc.
- The state space is represented as a combination of room temperatures and the energy level of the house. Temperatures and energy level are normalized to fall within the range [-1, 1].
- Rewards are computed based on the deviation of room temperatures from ideal values and the energy level of the house.
- The function step also handles periodic events such as adding or removing people from rooms and decays the exploration rate.

MDP

Observation space:

- T: Temperature
- EL: Energy level

MDP Formulation:

State Space (S): $S = \{(T, E) \mid T \in [16, 33], E \in [0, 100]\}$

Action Space (A): $A = \{(t_i, a_j) \mid t_i \in \{-3, -2, -1, 0, +1, +2, +3\}, a_j \in \{0, 1, 2, 3\}\}$

MDP

- No. of actions = 28

  The action space consists of discrete actions representing temperature adjustments and airflow adjustments. The total number of actions is 28.

  (7 temperature adjustments * 4 airflow adjustments)

Reward function

- In Energy Management System, we have two rewards.
    - Temperature Reward
    - Energy Reward

$$R(s,a) = R_{\text{temp}}(s) + R_{\text{energy}}(s)$$

$$R_{\text{energy}}(s) = \begin{cases} 1 & \text{if } E \geq E_{\text{threshold}} \\ -1 & \text{otherwise} \end{cases}$$

Reward function

- Temperature Reward:

$$R_{\text{temp}}(s) = -0.5 \times (|T - T_{\text{ideal}}|^2)$$

- $$T(s, a, s') = \begin{cases} 1 & \text{if } s' \text{ is the state resulting from applying action } a \text{ to state } s \\ 0 & \text{otherwise} \end{cases}$$

Agent

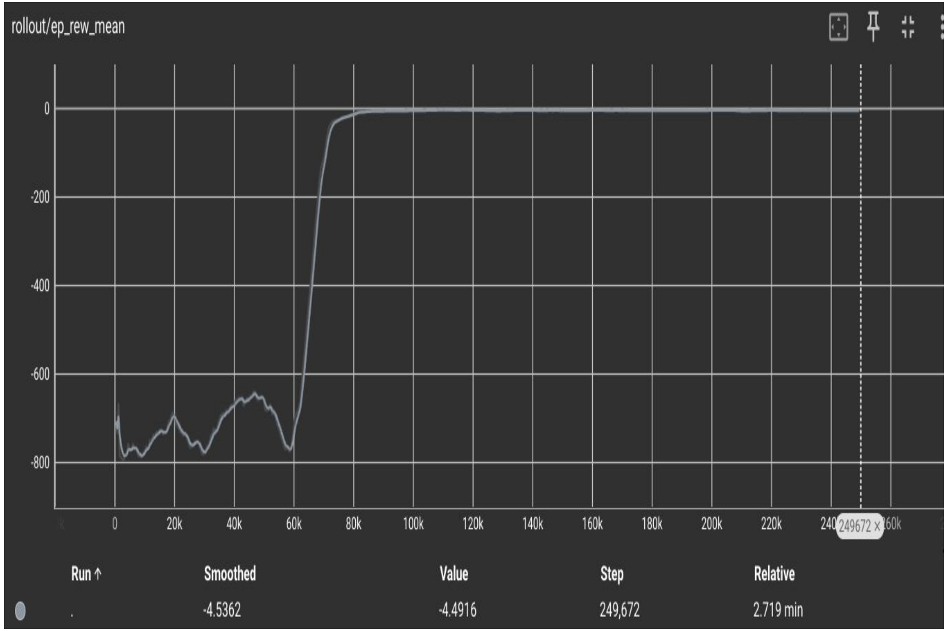
The Agents used in our project are:

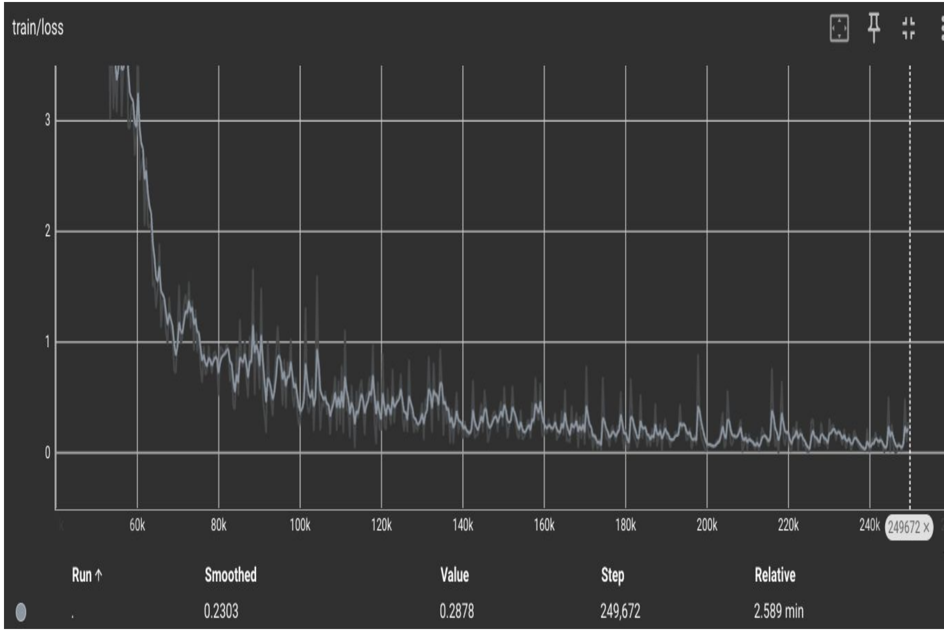
- **DQN**
  - Can learn complex strategies from high-dimensional inputs.
  - Utilizes deep learning for value approximation, suitable for discrete action spaces.
  - So this suits our environment which has discrete action spaces like temperature adjustments and airflow changes.
- **A2C**
  - Actor-critic algorithm that combines the advantages of both policy-based and value-based methods.
  - It is efficient and can be parallelized, making it suitable for both continuous and discrete action spaces.

- **PPO**
  - Stable and sample-efficient of policy gradient methods, works well for both discrete and continuous action spaces.
  - Effective in environments with complex dynamics and continuous action spaces like temperature and airflow adjustments.
- **SAC**
  - Maintains a balance between exploration and exploitation in continuous action spaces.
  - Effective for environments with high-dimensional observations and continuous action spaces like temperature and airflow control.

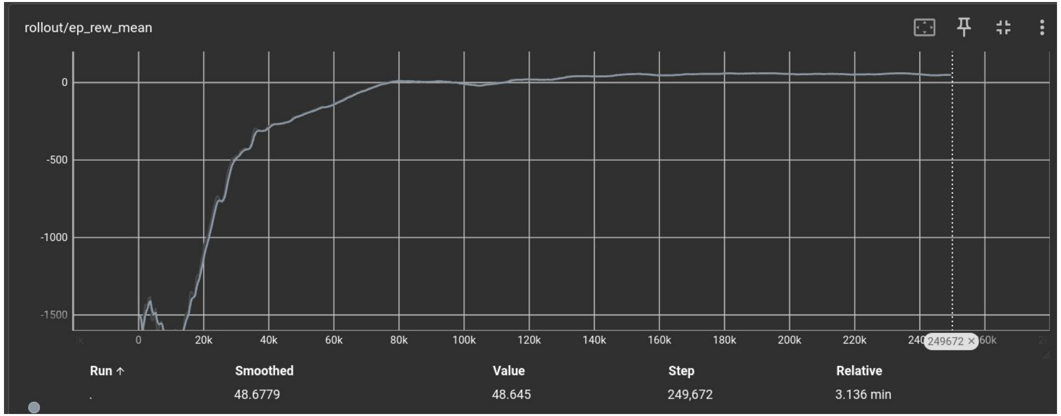
Results

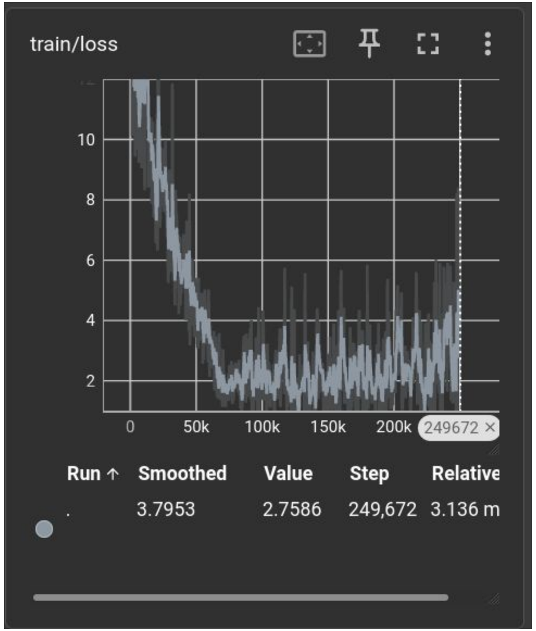
DQN Agent: House Env



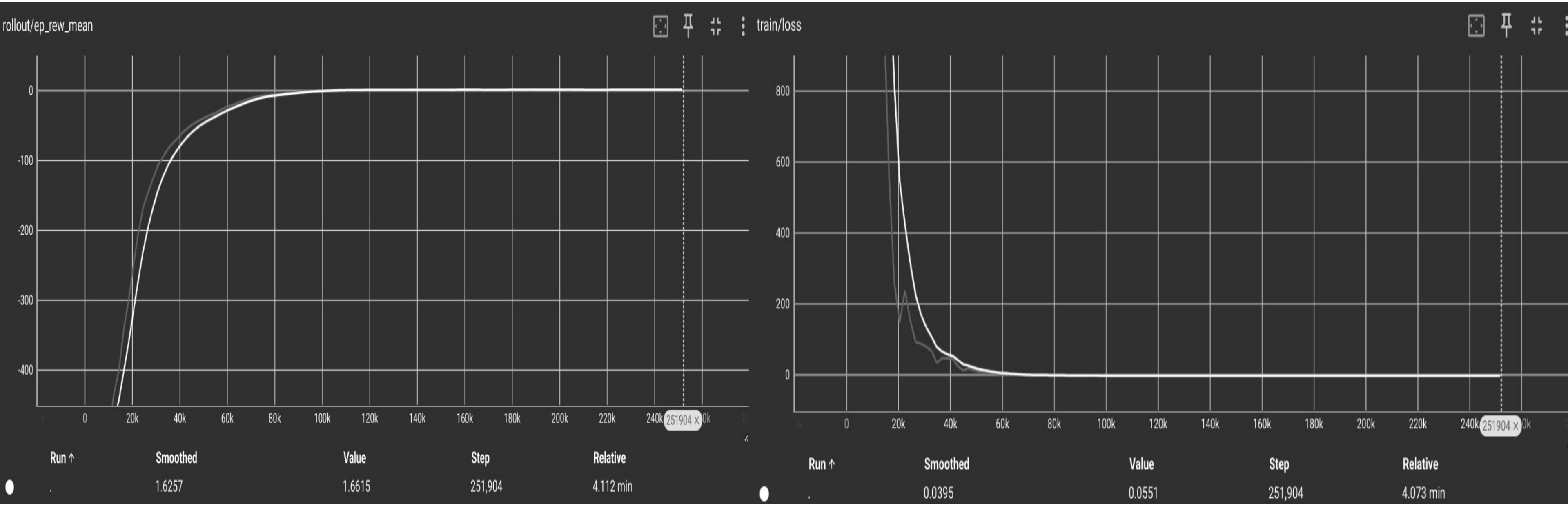


DQN Agent: Room Env

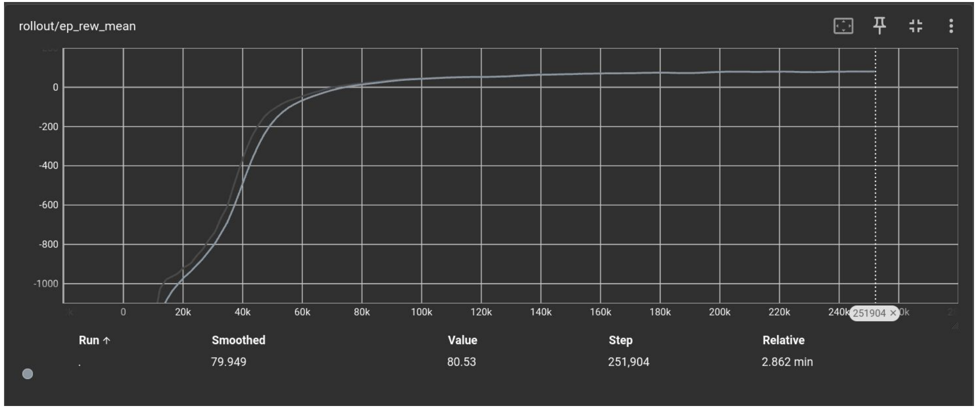


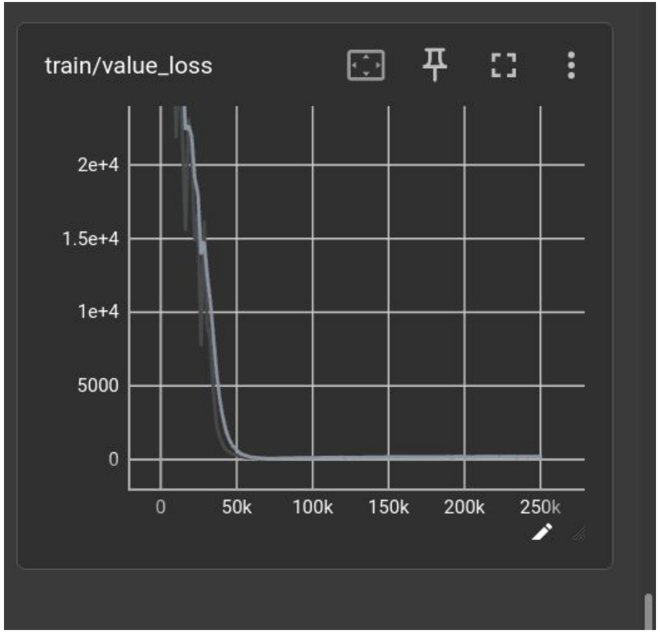


PPO Agent: House Env

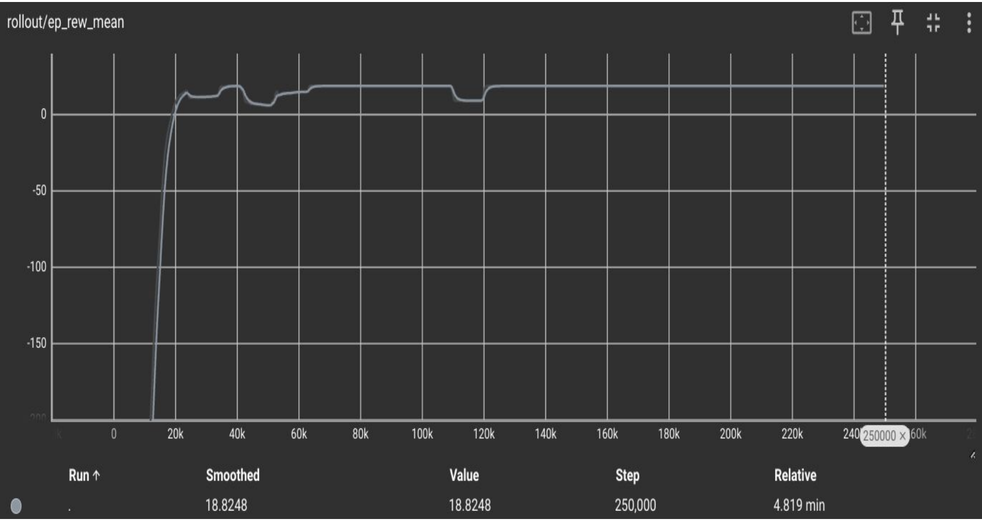


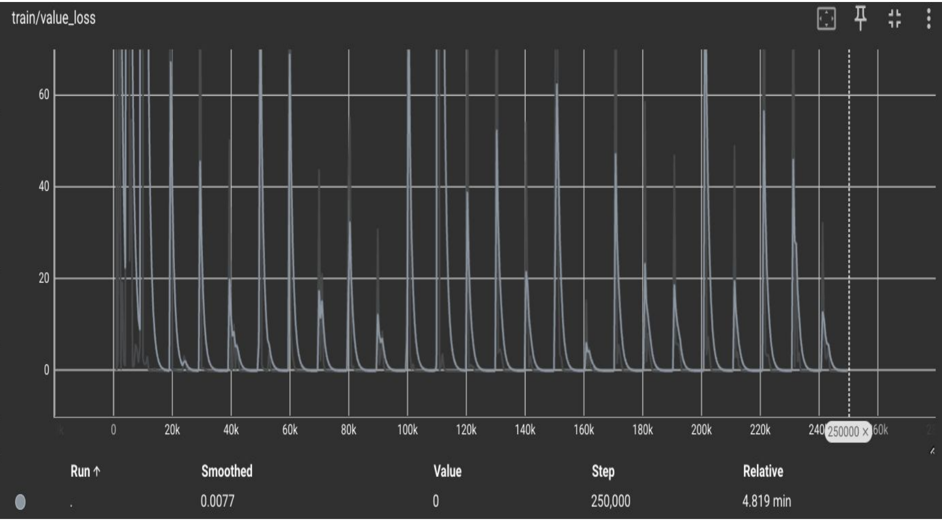
PPO Agent: Room Env



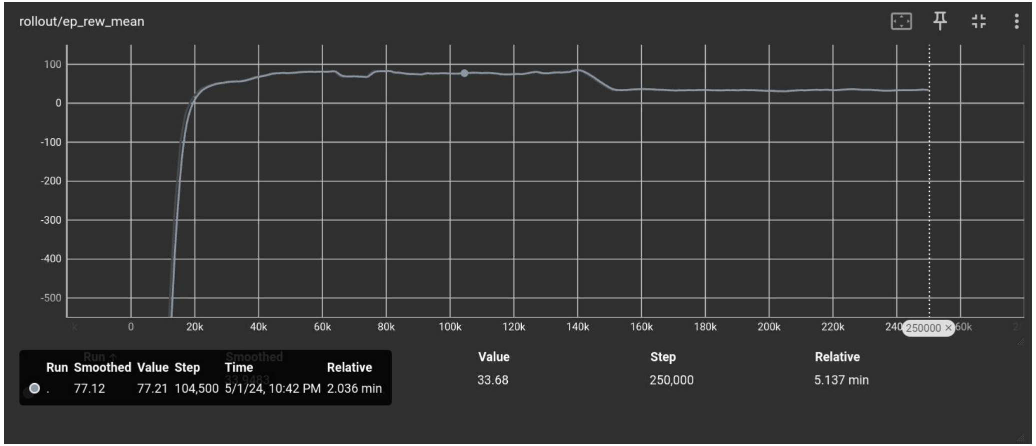


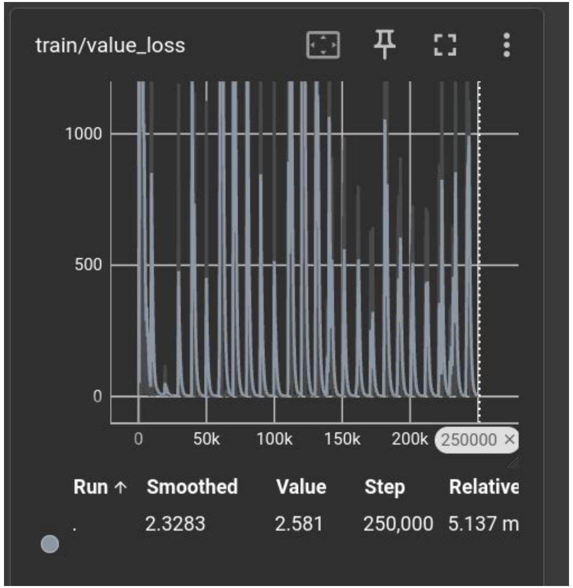
A2C Agent: House Env



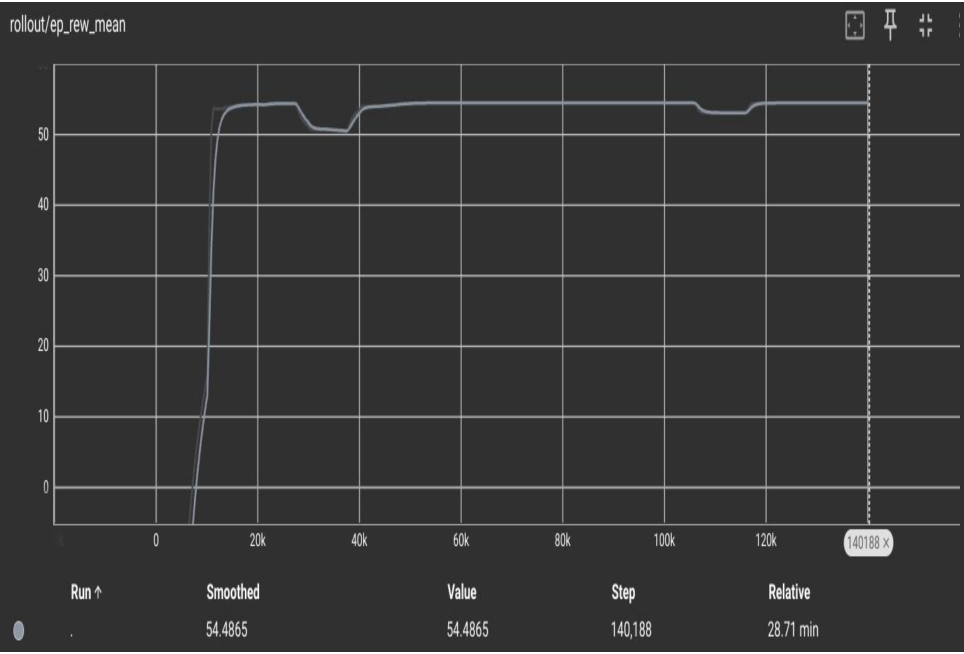


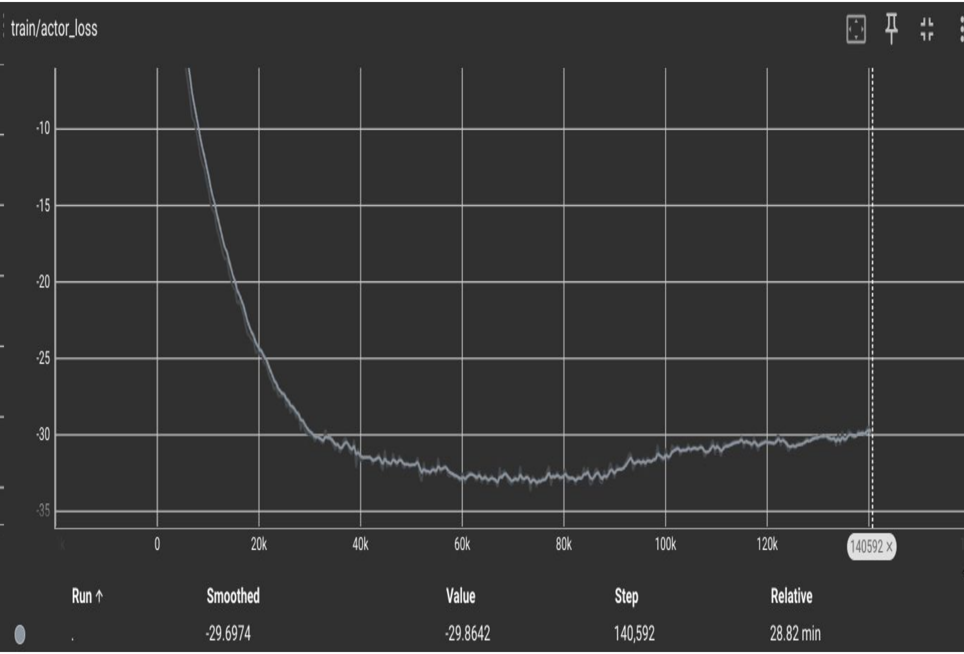
A2C Agent: Room Env



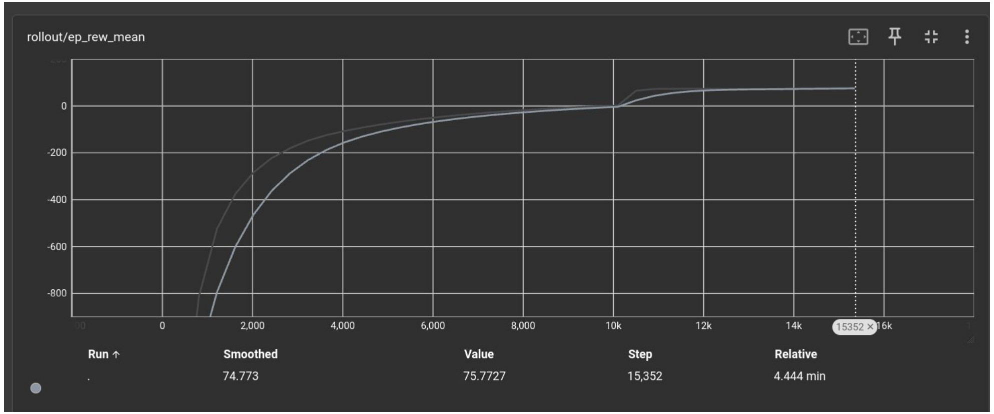


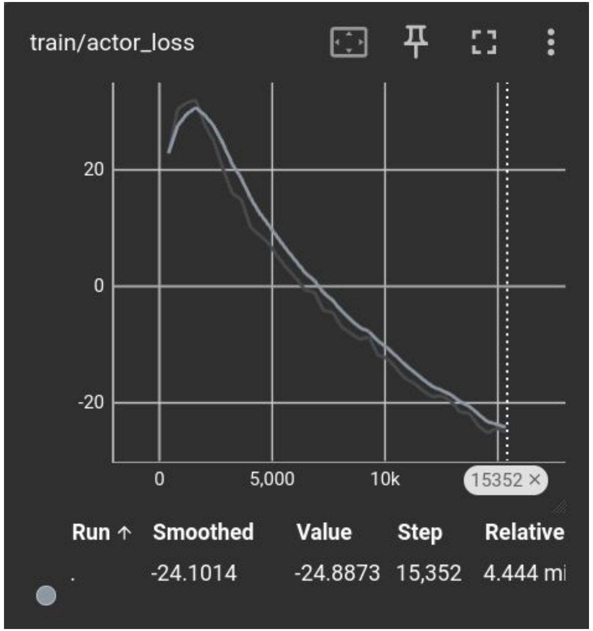
SAC Agent: House Env





SAC Agent: Room Env





Comparing various agents

- SAC is much slower compared to the other agents, but it was giving pretty accurate results, especially in the case of the House environment, where it performed very well, even though the training time was more than 45 minutes.

- A2C was a favourite of ours, as given it’s fast training time, which it achieved without compromising either the accuracy or stability.

- DQN was neither too good, nor too bad. It performed mediocre overall in our opinion. We believe this might be because, while the environment offers a lot to be explored, we didn’t have the time to train one specific model to its utmost, when refined the environment was our main goal.

- PPO offered a slight but non-negligible increase in performance over DQN in our results.

Future Scope

- We can make the House Environment more robust by removing the all the randomness we added to simulate a real house, and using actual environmental(weather) data, house parameters and other such details.

- Doing this, we can make an environment that can very closely resemble the real environment of a user’s home.

Thank You

Manish Reddy K

Harshadeep Donapati

Akhil Tavva