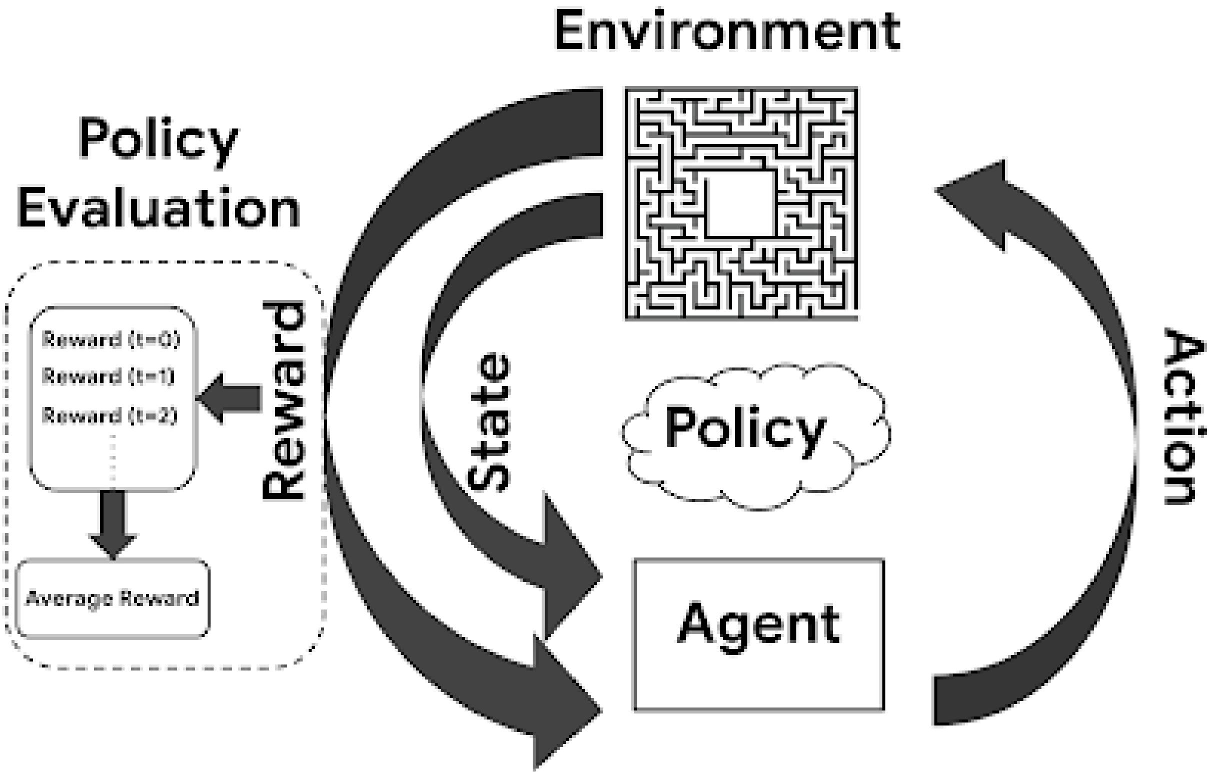
REINFORCEMENT LEARNING VIA SEQUENTIAL MODELING



Introduction

Reinforcement Learning Studying RL you are bound to come across the term the deadly triad. It is the three conditions such that if all are met, makes an agent extremely unstable. These 3 conditions are as follows:

1. Using function approximation (eg: using neural networks).
2. Using off-policy learning (Offline learning).
3. Using Temporal Difference Learning (Bootstrapping).

These 3 are good properties to use but we must avoid using all 3 together.

Function approximation is a must since it allows for generalisation when the state space is large.

Off-Policy learning allows you from running tedious and unfeasible simulations over and over and allows you to learn from a fixed bank from the data.

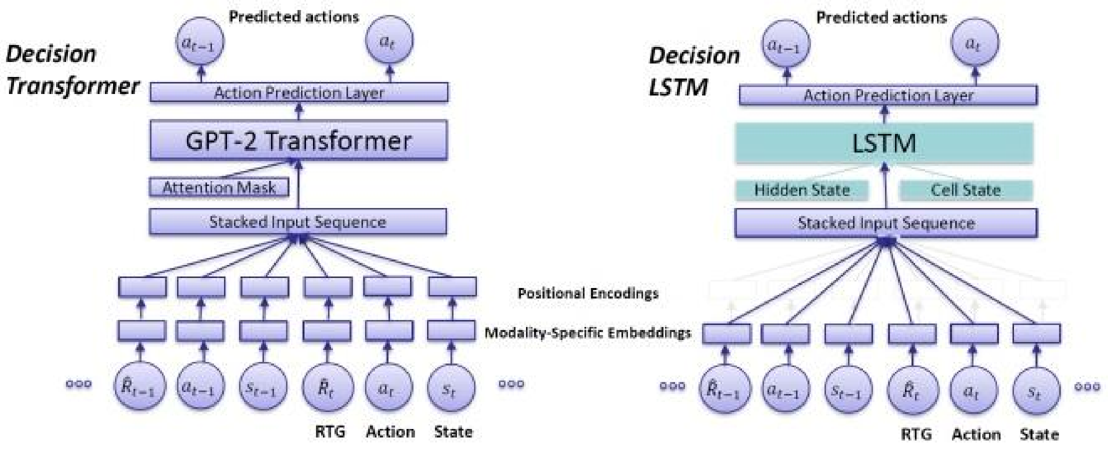
TD allows you to calculate the value functions on the go without completing the complete process and can allow for continual learning. TD normally have a discount factor which biases the agent towards short term goals which may not be ideal. Though an issue this is a necessity for bootstrapping and hence TD learning.

Newer works have looked at RL tasks through the lens of sequential learning and have developed a new method called decision transformer. It is a model which takes in a sequence of (R,S,A) tokens learn and predict the next step and reward. The paper 'Decision Transformer: Reinforcement Learning via Sequence Modeling' uses transformers to model these sequences. This leads us to wonder the obvious question of how other sequential models would perform. This is exactly what you would find out in the task.

Formally, you must do the following:

1. Read the paper mentioned above.
2. Implement their methodology but change the architecture with other sequential models other than transformers.
3. Train the model on the Hopper environment available on gymnasium.
4. Compare the results and give your thoughts on why this may be the case.

Decision Transformer vs Decision LSTM : Reinforcement Learning via Sequence Modeling



Parameters:
Learning rate : 1e-4
Weight Decay : 1e-4
Batch Size: 64
Warmup steps : 10000
Max Episode Length: 1000
Number of Episodes : 10
Max Training Iterations : 300
No of Updates Per Iteration : 100

Input Data:
Dataset : Medium
RTG Scale : 1000
Environment Name : Hopper-V3
RTG Target : 3600
Environment D4RL name : hopper-medium-v2

Results :
model loaded from:

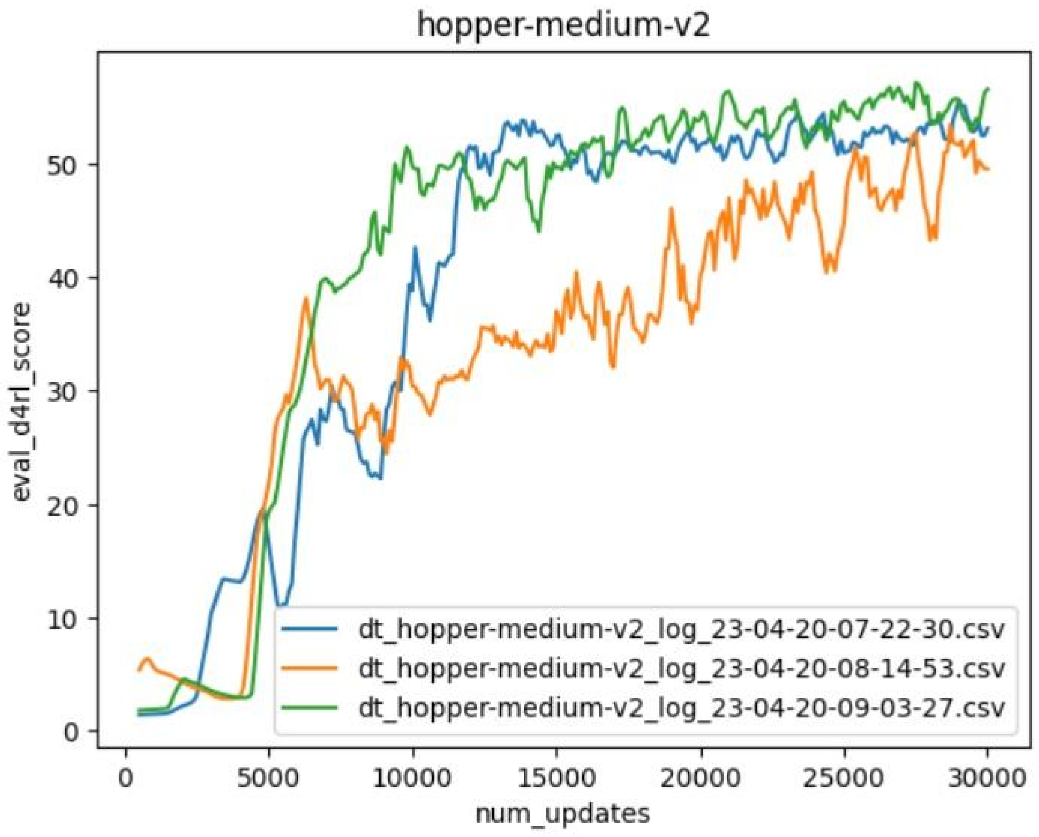
```
./dt_runs/dt_hopper-medium-v2_model_23-04-20-07-22-30_best.pt
{'eval/avg_reward': 1775.5411390889035, 'eval/avg_ep_len': 562.9}
normalized d4rl score:  55.17817014942317
model loaded from:
./dt_runs/dt_hopper-medium-v2_model_23-04-20-08-14-53_best.pt
{'eval/avg_reward': 1766.1849009695957, 'eval/avg_ep_len': 550.7}
normalized d4rl score:  54.89069034432147
model loaded from:
./dt_runs/dt_hopper-medium-v2_model_23-04-20-09-03-27_best.pt
{'eval/avg_reward': 1815.9646547640841, 'eval/avg_ep_len': 571.9}
normalized d4rl score:  56.42022323311339
============================================================
evaluated on env: Hopper-v3
total num of checkpoints evaluated: 3
d4rl score mean: 55.49636
d4rl score std: 0.66373
d4rl score var: 0.44053
============================================================
{'eval/avg_reward': 1572.3686380254937, 'eval/avg_ep_len': 501.0}
normalized d4rl score:  48.93549117279463
```

Execution:

The code completed execution in 170 minutes using 1 GPU resource on Google Colab

Plots:

```
dt_runs/dt_hopper-medium-v2_log_23-04-20-07-22-30.csv (300, 6)
dt_runs/dt_hopper-medium-v2_log_23-04-20-08-14-53.csv (300, 6)
dt_runs/dt_hopper-medium-v2_log_23-04-20-09-03-27.csv (300, 6)
```



CONCLUSION

DataSet	Environment	Decision Transformer with 100k updates	Decision LSTM with 30k updates
Medium	**Hopper**	**67.60**	**55.49**

As can be seen clearly, for the task of reinforcement learning on the Hopper Environment for the medium dataset, the D4RL mean value for the official version of Decision Transformer performs better than this implementation of the Decision LSTM . However the official results were obtained after 100k updates whereas the LSTM result was obtained after only 30k updates due to limits on Google Colab GPU resources.