Future Works

Going forward, we can explore these 2 areas in greater depths:

(1) A3C agents outperform our Actor-Critic agent in long term performance and speed of attaining these performances. Code analysis reviews three key differences between our Actor-Critic implementation from the A3C implementation:

1. **Parallelism** – A3C is capable of parallelism (1 to 16 agents), we have not implemented parallelism in our model.

2. **Advantage Function**

We uses the generic advantage function defined for Actor-Critic:

Advantage = Q\_t(s,a) - V\_t(s)

A3C implements the Generalized Advantage Estimation:

Advantage = Reward\_t + Gamma \* V\_t+1(s) - V\_t(s)

3. **Policy Entropy vs. Temperature**

The A3C and many subsequent papers (A2C and ACKTR) adding policy entropy to policy loss to encourage exploration, where policy entropy is defined as:

H(X) = -Sum P(x) log(P(x))

We on the other hand uses temperature to influence the policy distribution outputted by the softmax classifier:

action = torch.nn.functional.softmax(action\_head(x) /temperature)

(2) A more thorough study of how temperature and learning rate influence the agent’s long term performance (running reward) and the speed of attaining this performance. The study should be run based on controlled random seeds, so that we can better replicate an agent’s training. In our project, we frequently run into situations where we cannot replicate an agent’s peak performance when training it a 2nd time using the same hyperparameters.

(3) An investigation into how TBPTT can be optimized through its hyperparameter (chunk size) to achieve comparable performance as Backprop through Episode. The use of TBPTT greatly reduces the GPU memory demand, which allows a research to run more parallel experiments on a GPU-equipped workstation.