**Implementing policy gradient using “differential training”**

**Milestone Report**

Hossein Najafi, Roham Ghotbi

**Progress since proposal**

Combing through the literature and relevant articles, we have successfully worked out the correct method of implementation for differential training, in terms of loss functions to be considered, gradient descent schemes and how to translate these into tensor implementations.

The core of this method is to train a critic that takes in two (state, action) as input and outputs the difference of their respective Q’s. A subsequent actor will then use this critic to select the best action, until the critic reaches a stationary state at which point training is then complete. The article[1] we used as reference, implemented an “argmax” policy iteration to train the actor from the critic at each step, but we decided to replace this training with gradient step on the loss function instead, which is more straightforward to implement.

Since the (state, action) pairs are now simultaneously differentially evaluated using one single network, the literature argues that this translate into more robustness in the face of noise and random disturbances, which we will evaluate and review in this work. We have implemented the core of this method in **pytorch** and we have so far been able to evaluate and test it on 2 different environments in **Mujoco** (Pendulum-v0 and LunarLanderContinuous-v2) and our “average eval return” plots for both our test environments are shown below.

Chart, line chart

Description automatically generated

Figure 1. Performance of differential training on LunarLandercontinuous-v2 (batch=32, lr=1e-3)

Chart, line chart

Description automatically generated

Figure 1. Performance of differential training on Pendulum-v0 (batch=32, lr=1e-3)

**Future work**

Our next focus is to finalize the algorithm, introduce disturbance in the system and evaluate the performance. We will also perform control experiments with conventional methods to compare with differential training.

**Reference**

[1] Bertsekas, D. (1998). Differential Training Of Rollout Policies*,* Proc. of the 35th Allerton Conference on Communication, Control, and Computing.