GSOC Application - GFOSS

Name : Sanket Kumar Thakur

About Me : Resume

Stay connected :

Title : Extend deepbots to support stable-baselines and implement gym-style default Reinforcement Learning environments

Contribution :

- Develop RL based environments which are based on openAI gym scenarios (https://gym.openai.com/envs/#classic_control).

    1. MountainCarCountinous-v0 which can be used with BB-8 robot. Discrete action based mountaincar have already been resolved here(pitescape). Similar environment can be used for the continuous action space. This environment can be solved with either DDPG or PPO. Reference solution using CEM.

    2. LunarLander-v2 using the Mavic 2 robot. Reference solution using DQN

    3. Acrobat-v1 / Pendulum-v0 using the IPR / IRB robot / PUMA (not sure). If we can get the joints to work independently like a pendulum. Reference solution using DDPG.

    4. Since both the environments (2, 3) have a discrete action space, I want to add the functionality of policy based algorithms such as REINFORCE - based on Monte Carlo method of MDP (optional, if time permits) , CEM(Cross Entropy Method) [1] and also DQN [2].

- The environments should be enhanced to support following categories :
    - **continuous state, discrete action space** - LunarLander, Acrobot
    - **Continuous state - action space** - MountainCarContinous-v2
    - **Discrete space - action** - Create a new simple testbed env different from the gym environment. - Can I create Taxi-v3 using ALTINA and solve it using simple q-learning ??

    - Other ideas :
        - Snake : The environment can be a simple snake game using salamander with the position of the snake's head, its length and its distance (Euclidean) from the bait constituting as state observations and the actions being turn left, up, down or right (4). The reward is provided for most bait eaten. - Continuous state, discrete action.

- Properly document the use of environment and algorithms to support users in their own implementation. Also, create easy-to-use methods and classes to help others tweak with the hyperparameter optimization of the algorithms and states of the environment.

- **Tech Stack and tools :**
    - **Python**
    - **Pytorch**
    - **Pipenv, mypy**

Timeline :

- **Community Bonding weeks :** Study the emitter-receiver scheme wonderfully described here and get used to the web bots framework. Develop a basic environment or the one mentioned in the tutorial using different sensors. Communicate doubts regarding the framework with mentors and create a blueprint for the environments that are to be targeted. Also discuss a testbed environment that can be built without referring to the gym and targeting the discrete space-action space.

- **Week 1 - 2 :** Work on the first environment project and add controllers to the robots. Create the robot controller and supervisory scripts for robot control and states.

- **Week 3 :** Decoding bugs and checking the code to make sure the agents work properly and the states are properly transmitted through the framework.

- **Week 4 - 5 :** Work on the second and third environment project along with their necessary scripts.

- **Week 6 :** Submit the mid level evaluation results and progress.

- **Week 7 :** Spare week if delays from week 4 spill-over. Discuss other stable-baseline methods(suggesting DQN or REINFORCE) which can be added to the codebase. Develop a concrete design for the custom testbed. Also, consider adding an infrastructure for hyper parameter optimization. We can decide if to work on adding a baseline method or instead on the hyperparams optimization using ray.

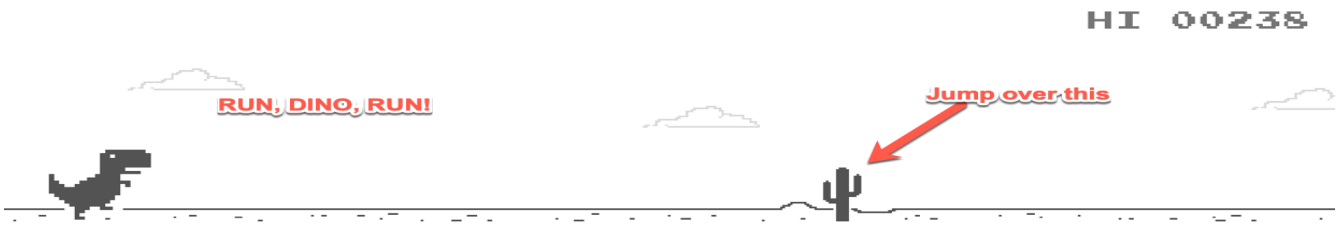
- **Week 8 - 10 :** Work on adding stable-baselines or creating an infrastructure in deepbots for hyper params optimization which can be integrated into the workflow, as discussed from week 7. Also, in parallel develop the custom testbed.

- **Week 11 :** Spare week in case of work getting delayed, solving bugs and decoding the codebase.

- **Week 12 :** Prepare and submit the final evaluation results.

Am I good enough ?

- I have previous experience in building a python environment and then using it to train a RL agent. Dino AI - where I created a simple clone of google chrome game.



So, I have 5 value based state objects and 3 actions (0 - nothing to do, 1 - jump, 2- crouch, it might vary for two agent scripts). The results are here. https://www.youtube.com/watch?v=mrzyq8SkX0A

- I have worked on the mentioned algorithms using openAI gym environment and maintain a github repo while applying the algorithms on different gym based environments. Check it here. https://github.com/sanketsans/openAIenv

- I am also currently in top leaderboards for some openai gym environments.

Why am I doing this ?

- I am really excited about Reinforcement Learning and especially its applications in human-level intelligence in games. I was fortunate enough to work with RL during my internship and since then I try to learn new algorithms and implement them in a gym based environment or other simple environment which either I find on github or I create them.
- I recently found out about deepbots and it seems exciting to me and since it is still growing, I believe it can give me an opportunity to work on different RL environments while working on open-source.
- Finally, GSOC is a really great opportunity to indulge myself into open source contribution and connect with mentors and like-minded people to establish connections and obviously help the community.
- Besides, I will want to stay with the organization in future as well.

Reference:

1. Shie Mannor, Reuven Rubinstein, and Yohai Gat. 2003. The cross entropy method for fast policy search. In *Proceedings of the Twentieth International Conference on International Conference on Machine Learning* (*ICML'03*). AAAI Press, 512–519.

2. Mnih, V., Kavukcuoglu, K., Silver, D. *et al.* Human-level control through deep reinforcement learning. *Nature* **518,** 529–533 (2015). https://doi.org/10.1038/nature14236
3.