Project Proposal

Group Members:
Xupeng Zhu, zhu.xup@husky.neu.edu, 001814074
Zimou Gao, gao.zim@husky.neu.edu, 001448243

Problem Description:

- Classical Control: Cartpole Environment (https://gym.openai.com/envs/CartPole-v1/)
    - States:
    - Action: Move left or right
    - Rewards:
- Atari games: Break Out (https://gym.openai.com/envs/Breakout-v0/)
    - States
    - Action
    - Reward
- Robotics: Pybullet KUKA Pick and Place (https://pybullet.org/wordpress/) (XPZ)
  Train the RL agent to control the KUKA manipulator to pick object by depth image.
    - States: Simulated depth image that contains the position of the desk and the object
    - Action: discrete: move along the x-axis, y-axis, z-axis, angular rotation, with a discrete value and close gripper; continues: move along the x-axis, y-axis, z-axis, angular rotation, with continues value and close gripper.
    - Reward: if failed grasping reward = -1, succeed reward = 1.

Algorithms:

- Q-learning
- DQN (tabular)
- Double DQN (tabular):
- Dueling DQN (tabular):
- A2C (Policy Gradient):(XPZ) A2C is Advantage Actor Critic in short. Actor Critic algorithm contains policy network that will generate actions (actor) and value network will evaluate these actions (critic). They collaborate to improve the learning speed as well as stability. The Advantage Actor Critic algorithm incorporated the advantage function in the value network that can make use of the training data to speed up the training.

- DPG (Policy Gradient):(XPZ) DPG is Deterministic policy gradient in short. DPG utilizes policy gradient to generate continues action value. To solve exploration - exploitation problem, DPG is off-policy and train on an exploratory behaviour policy. DPG The deterministic property outperforms the stochastic policy gradient algorithms in high-dimensional action spaces.

Results:

Results to expect:

- Learning curve
- Training time
- Average reward

Comparisons:

The risks: (XPZ)

For DQN, DDQN, Deuling dqn: since it may use convolutional neural networks, it tends to be have to find the proper networks structure to approximate Q function. It may also takes too long to train on complex problems. Moreover, these algorithms are for discrete control, they may fail in continues problem (though discretized).

For A2C and DPG: the implementation maybe to complex and easy to have bug. The parameters like learning rate may be hard to tune. The policy gradient may lead to instability of policy that will ruin the trained policy.

How to deal with risks: (XPZ)

For DQN, DDQN, Deuling dqn: we will try different neural networks structure to find the proper structure. During training we can use GPU to speed up RNN training. If these algorithms fail in continues problem, we can try more fine-grained discretization.

For A2C and DPG: we will apply the algorithm in simple environment to test the functionality of each part of the algorithm first then apply in complex environment. Simple environment can be a practice for tuning the parameters. Some contraine can be applied to stabilize the policy gradient.