ECE276 Reinforcement Learning Final Project

Advantage Actor Critic(A2C) with Experience Replay

Chun Hu[†] Po-Jung Lai[†] Chih-Hui Ho[*]
Department of Electrical Engineering [†] Department of Computer Science [*]
University of California San Diego

Contact Information:
Email: chh281@eng.ucsd.edu
polai@eng.ucsd.edu
chh279@eng.ucsd.edu

Abstract

We propose a simple and lightweight framework for deep reinforcement learning that uses synchronous gradient descent for optimization of deep neural network controllers with experience replay. We incorporate synchronous, deterministic variant of reinforcement learning algorithms, named A2C, with experience replay and show that parallel actor-learners have a stabilizing effect on training allowing the algorithm to successfully solve games in OpenAI gym. Moreover, different experience replay buffer size and sampling techniques are compared. Our result outperforms the baseline A2C without experience replay.

1 Introduction

In the literature of reinforcement learning, several methods are proposed to improve the policy and those methods can be coarsely categorized into 2 categories. The first category finds the optimal policy indirectly through the surrogate optimal value function, while the other methods, or policy iteration, optimize the policy directly. Actor-critic [1] method is a well-known method that generalizes policy iteration. It iterates between the policy evaluation process and the policy improvement process. 2 modules, an actor module and a critic module, are interacting with each other. The actor module aims at improving the current policy, while the critic module evaluates the current policy.

A recent paper advantage actor-critic method [2] discussed an alternative way to train the system by using synchronous gradient descent for optimization of deep neural network controllers and show that the stability of the network is improved. In this final project, the advantage actor-critic(A2C) method is implemented. Several Atari games and traditional control problems were experimented using the algorithm. In addition, experience replay technique is utilized to boost the performance of the network.

2 Related work

Advantage actor-critic (A2C) method [2] proposed to train the network in synchronous way and apply to different RL learning algorithms, including SARSA, Q-learning and actor critic. The motivation of proposed method is to solve the instability when training the network in the same thread, due to the high correlation between the training data. Although adding experience replay techniques can alleviate the problem, it constraints the training procedure to be off-policy and the replay buffer will increase significantly as more experience is needed, which results in high memory usage.

3 Project Idea

The baseline implementation is to replicate the A2C algorithm. Since the A2C only depends on the on-policy update, the algorithm discard all the trajectories after updating. To further extend the stability and capability of the network, we implement the algorithm Sample Efficient Actor-Critic with Experience Replay[3] by adding the experience replay to the original A2C algorithm. We wonder that the old trajectories may be still useful as for updating the networks. Hopefully, We will show that utilizing the appropriate size of experience replay buffer will stabilize and boost the performance of the network.

4 Evaluation Metric

Assume the network is trained with T threads. All the returns for each thread are recorded during the training procedure. When the training is done, the returns of each trajectory for each thread is sorted according to the time it is generated. The typical running average methods is applied on the sorted reward. The results are shown in Figure 1. The performance of our result is comparable to the state-of-the-art result

5 Algorithm

Algorithm 1 Advantage actor-critic with experience replay

```
//Assume global shared parameter vectors $\theta$ and $\theta_v$ and global shared counter T = 0
// Assume thread-specific parameter vectors $\theta'$ and $\theta_v$
  Initialize thread step counter $t \leftarrow 1$
  for $T < T_{max}$ do
    Reset gradient: $d\theta \leftarrow 0$ and $d\theta_v \leftarrow 0$
    Synchronize thread specific parameters $\theta' = \theta$ and $\theta'_v = \theta_v$
    $t_{start} = t$
    Get state $s_t$
    while not terminate or $t - t_{start} < t_{max}$ do
      Perform $a_t$ according to policy $\pi(a_t|s_t;\theta')$
      Receive reward $r_t$ and new state $s_{t+1}$
      $t \leftarrow t+1,$
      $T \leftarrow T+1$
    end while
    Add the trajectory into replay buffer
    $X = \begin{cases} 0, & \text{for terminal} s_t \\ V(s_t, \theta'_v), & \text{for non-terminal} s_t \end{cases}$
    for $i \in t-1, \ldots, t_{start}$ do
      $R \leftarrow r_i + \gamma R$
      Accumulate gradients wrt $\theta'$ : $d\theta \leftarrow d\theta + \Delta_{\theta'} log\pi(a_i|s_i;\theta')(R - V(s_i;\theta'_v))$
      Accumulate gradients wrt $\theta'_v$ : $d\theta_v \leftarrow d\theta_v + \partial(R - V(s_i;\theta'_v))^2/\partial\theta'_v$
    end for
    Replay the mini-batch and update the network
  end for
```

6 Results

6.1 Experience Replay Buffer Size comparison

Different replay buffer sizes are experimented in this experiment, as illustrated in the first column of Figure 1. With appropriate replay buffer size (around five to twenty), the performance with replay buffer outperforms the baseline network. It is observed that the received reward is sensitive to the replay buffer size. The result shows that large(> 50) buffer size will harm the performance significantly, because more unrelated past experiences are likely to be sampled.

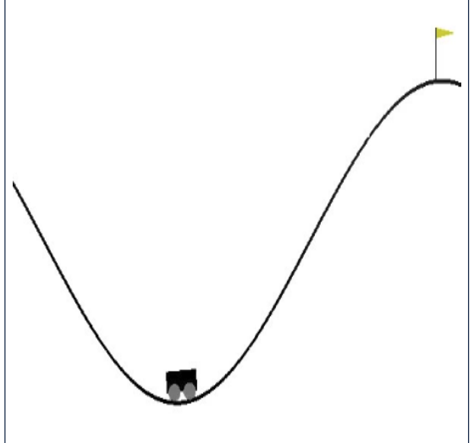
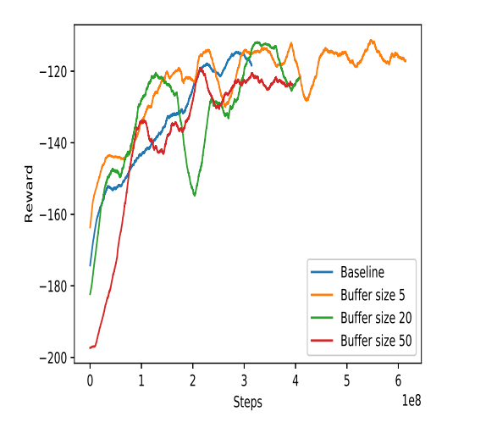
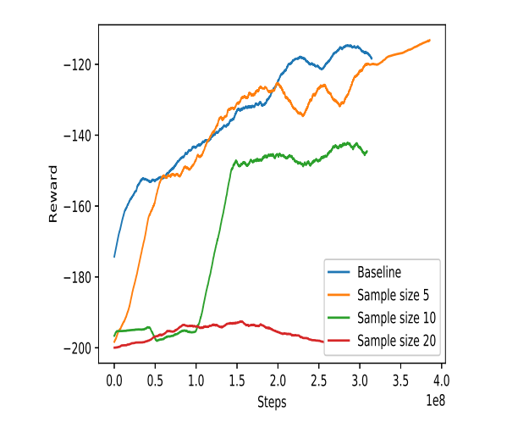
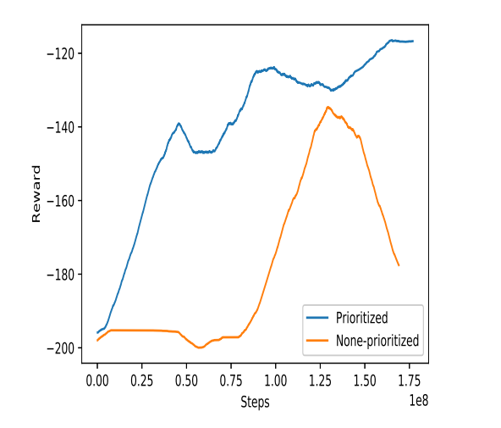
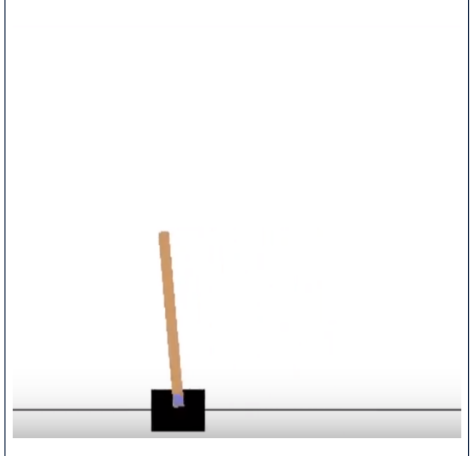
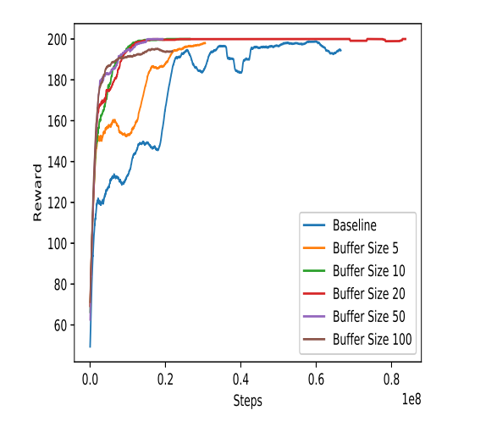
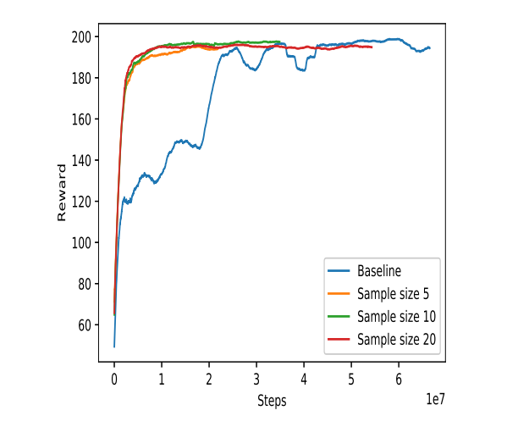
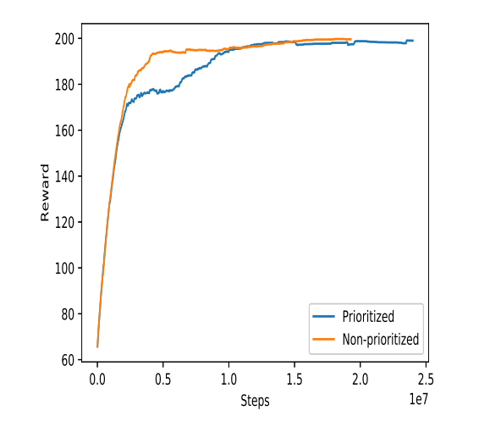
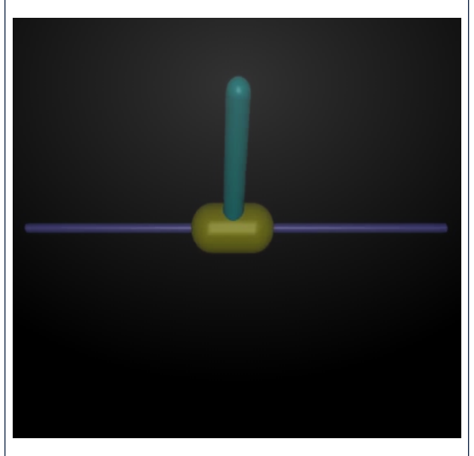
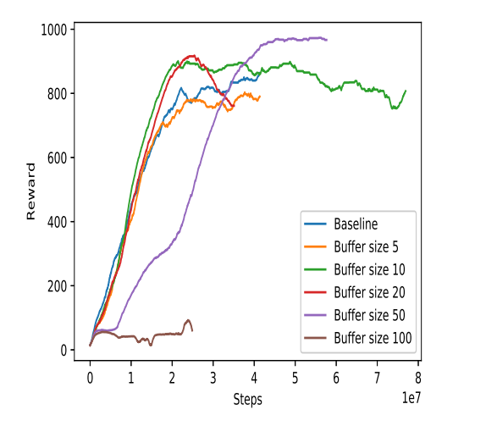
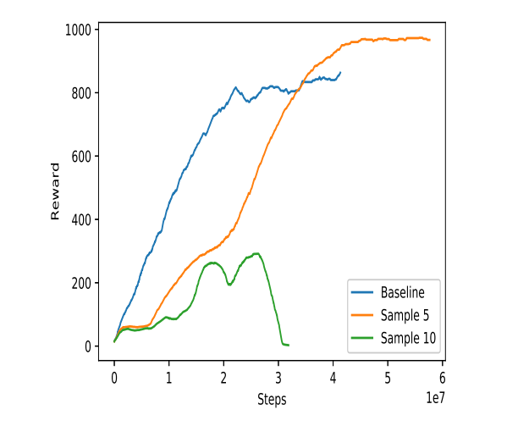
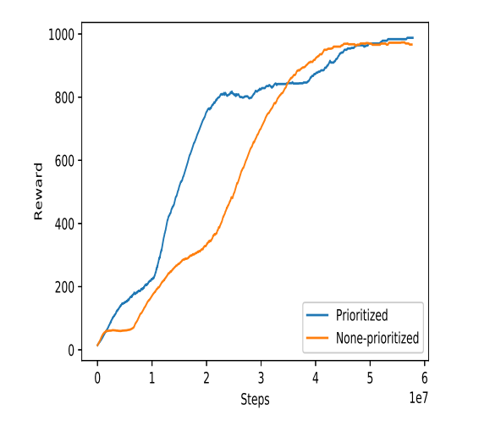
6.2 Experience Replay Sample Size comparison

This experiment shows the relationship between reward and sample size under fixed buffer size, as shown in the second column of Figure 1. As the sample size increases, more past experience are sampled. Since some of the old trajectories deviate too far from the trajectory generated from current policy, sampling too many past experience will degrade the performance of the network.

6.3 Prioritized vs Non-prioritized Experience Replay

This experiment investigates the effect of prioritizing the importance of different trajectories. As demonstrated in the third column of Figure 1, sampling the trajectory with higher reward tends to boost the performance of the network. In addition, prioritizing the trajectories in replay buffer speeds up the convergence of the network.

Figure 1: Reward plots for different environments under different experience replay settings

Game	Sample Image	Buffer size comparison	Sample size comparison	Prioritized comparison
MountainCar				
CartPole				
Inv. Pendulum				

7 Conclusions

In this project, we demonstrate that the algorithm, A2C with experience replay, can make the convergence speed much faster than the baseline model (A2C). The conducted experiment shows the effect of different buffer size and sample size settings. With adequate experience replay, our algorithm can surpass the performance from the baseline.

References

[1] V. R. Konda and J. N. Tsitsiklis, "Actor-critic algorithms," in *Advances in Neural Information Processing Systems 12*, S. A. Solla, T. K. Leen, and K. Müller, Eds. MIT Press, 2000, pp. 1008–1014. [Online]. Available: http://papers.nips.cc/paper/1786-actor-critic-algorithms.pdf

[2] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," *CoRR*, vol. abs/1602.01783, 2016. [Online]. Available: http://arxiv.org/abs/1602.01783

[3] Z. Wang, V. Bapst, N. Heess, V. Mnih, R. Munos, K. Kavukcuoglu, and N. de Freitas, "Sample efficient actor-critic with experience replay," *CoRR*, vol. abs/1611.01224, 2016. [Online]. Available: http://arxiv.org/abs/1611.01224

[4] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," *CoRR*, vol. abs/1801.01290, 2018. [Online]. Available: http://arxiv.org/abs/1801.01290

Acknowledgements

We thank the assist from the instructors in ECE 276C for setting up the GPU cluster and providing the suggestion for the final project.