Contents

Project 3

Collaboration and Competition

Deep Reinforcement Learning Udacity Nanodegree

Task

Play tennis. It is cooperative task because game is socred based on the number of exchanges. Every time agent hits ball to the opponents corner game reward is increased by 0.1 point.

Input

Vector of size 24.

Rewards

Based on the number of exchanges. 0.1 point per ball exchange.

Goal

Reach average reward of at least 0.5 points per episode - which means at least 5 exchanges.

Solution

We used code from previous project 2, Deep Deterministic Policy Gradient learning repo link.

Tennis problem

Parameters

- `mx` - number of maximal training iterations `10000`
- early stopping - if average score was not improved in the last 200 iterations training will stop.

Modifications from Continuous Control project

Key modification was in replay buffer and learning process.

Memory storing changes

We modified `ReplayBuffer.sample()` method. The code change allows access to both agents data.

```
class ReplayBuffer:
    """Fixed-size buffer to store experience tuples."""
    ...
    def sample(self):
    ...
        states = torch.from_numpy(
            np.stack([e.state for e in experiences if e is not None], axis=1)).float().to(device)
```

Learning process

Thanks to the replay buffer modifications we were able to change the learning. We have access to states and other parametes of both agents.

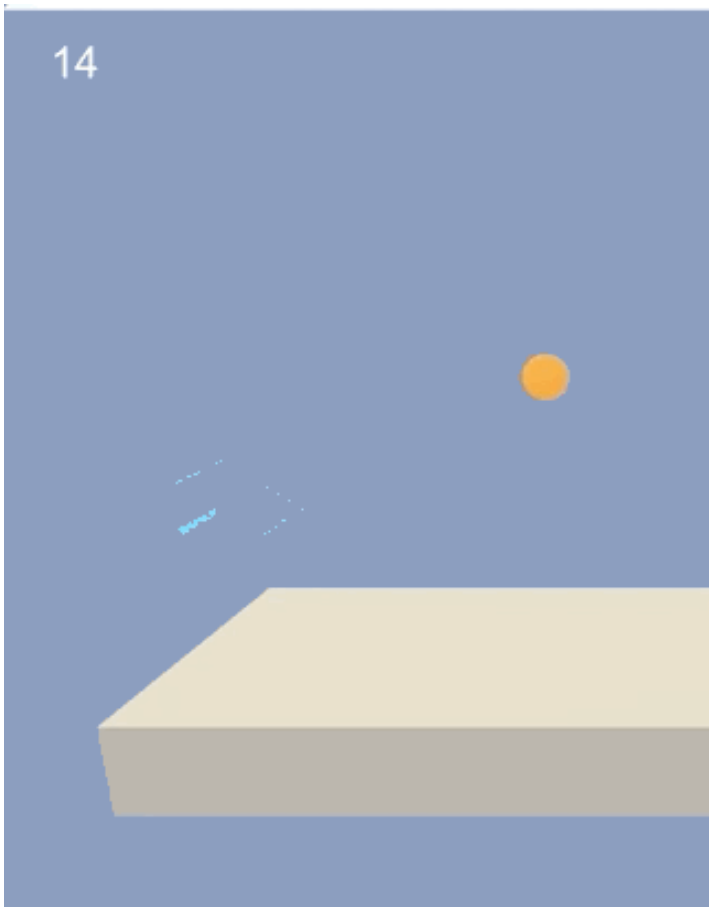
```
    def learn(self, experiences, gamma):
        """Update value parameters using given batch of experience tuples.

        Params
        ======
            experiences (Tuple[torch.Variable]): tuple of (s, a, r, s', done) tuples
            (all of the experiences are actions of both agents stacked)
            gamma (float): discount factor
        """
        states, actions, rewards, next_states, dones = experiences

        for agent in range(self.num_agents):
            opponent = (agent + 1) % 2
            ...
            Q_targets = Q_targets_a + rewards[opponent]
```

Rewards of opponent (in fact co-player) were added to the targets - because the task was cooperative.

Training

We tried multiple changes to the learning process but two mentioned above were key for successfull training. Neural network architecures are the same as in `ddpg-bipedal` for actor 1st layer contains 200 neurons and 2nd layer has 100 neurons 3rd layer has 2 neurons. Ativation function is hyperbolic tangens because action is vector of size 2 and values are continuous from -1 to 1. For critic base configuration is the same as for actor with different 3rd layer which contains only 1 neuron and linear activation function. We have experimented with neural network containing one more hidden layer. However it seems that learning is faster with fewer neurons and simpler architecture. Model was saved every time new rolling average maximum was reached. If no new rolling average maximum was reached in 200 episodes traning was terminated.

To reproduce achieved results parameters you can run `python3 train-tennis.py` (assuming all of the `requirements.txt` are full-



filled). Image showing solution to the problem. Paddles are hitting the ball.

- Objective was reached in 1494 episodes.
- Maximum rolling average was reached in 2099 episodes.
- Max rolling average: 1.406
- Max score: 2.7

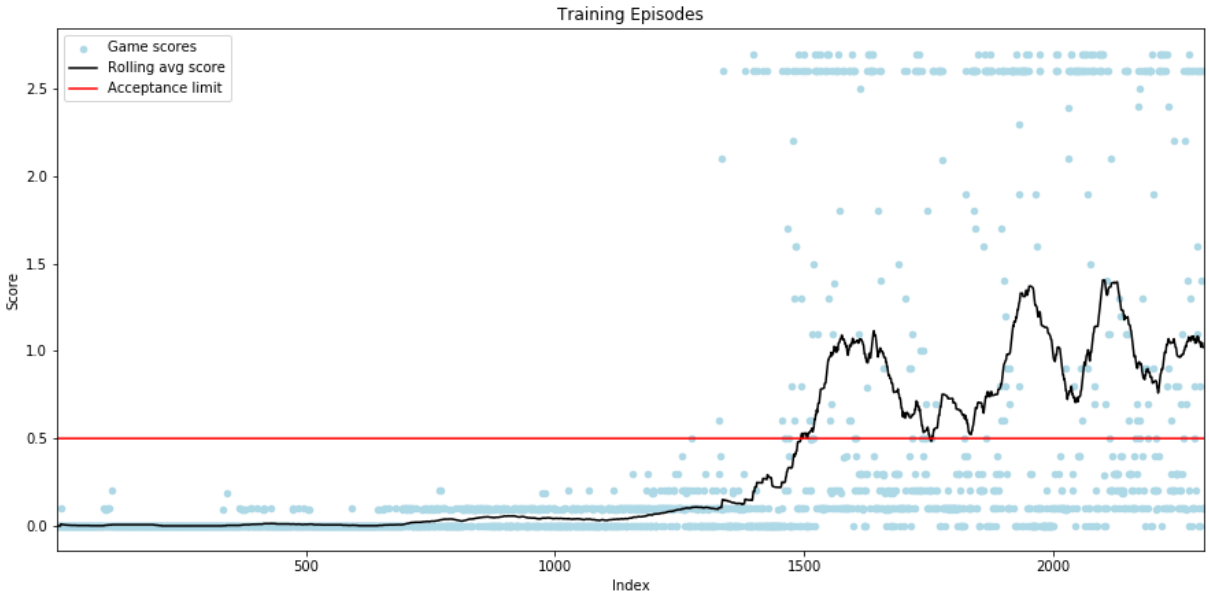
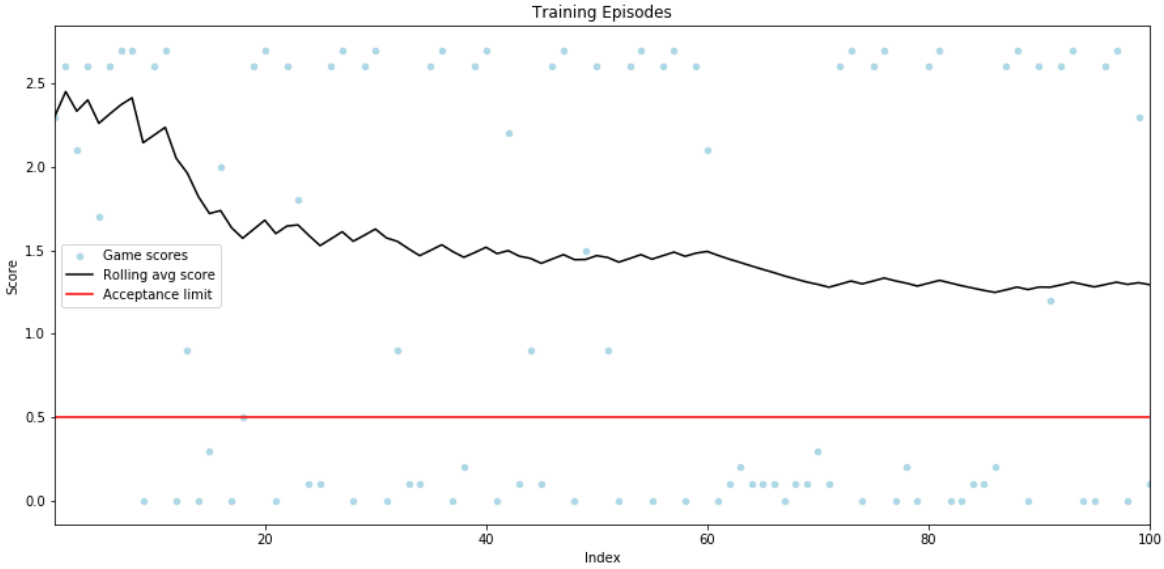


Figure 1: allt text

Test

- Test run with model weights loaded from `model/weights_local.torch` and `model/weights_target.torch` weights for cirtic are stored and loaded as well.
- To test included model you can run `python3 test-tennis.py` it will generate file data-test.csv with performace data recorded.
- Rolling average after 100 test episodes: 1.409



- Max score: 2.7

Conclusions

To improve training score we would have to understand why in some exchanges no paddle or at most one paddle hits the ball. This seems to be consistent pattern whenever ball is hit at least two times the game usually reaches high score above 0.5 (target). Once game is stable agents can keep the ball in the game for quite a few exchanges. It is important for the agent to somehow sabilize the ball. The first few hits are most important.

We can observe the graph and see two clear clusters one above 2.5 and the other around 0 points. Any training changes need to consider the starting conditions.