CSC411: Assignment #4 Bonus

Due on Monday, April 4, 2018

Zhiyan Deng, Xin Jie Lee

April 4, 2018

Prefix

The code for this project is written in Python 3

Problem 1

***Part 1**: X and O*

In project 4, the machine learning agent is always the first player and its opponent, a random agent, always goes next. In a game of tic-tac-toe between two random players, the player who goes first will often win approximately 60% of the games played. Hence, a more balanced approach will be to let our machine learning agent go first only 50% of the time during training. We hope that doing so will allow our learning agent to learn better policies and boost its win rate performance. The rewards structure was modified as well:

```
    """
    returns = []
    time_step = len(rewards)
    for i in range(time_step):
        curr_ret = 0
        for j in range(i,time_step):
            curr_ret += rewards[j] * gamma**(j-i)
        returns.append(curr_ret)
    return returns
```

Using a neural network with 128 hidden units to track the policies, the model was trained for 76,000 episodes and the training curve is depicted below:

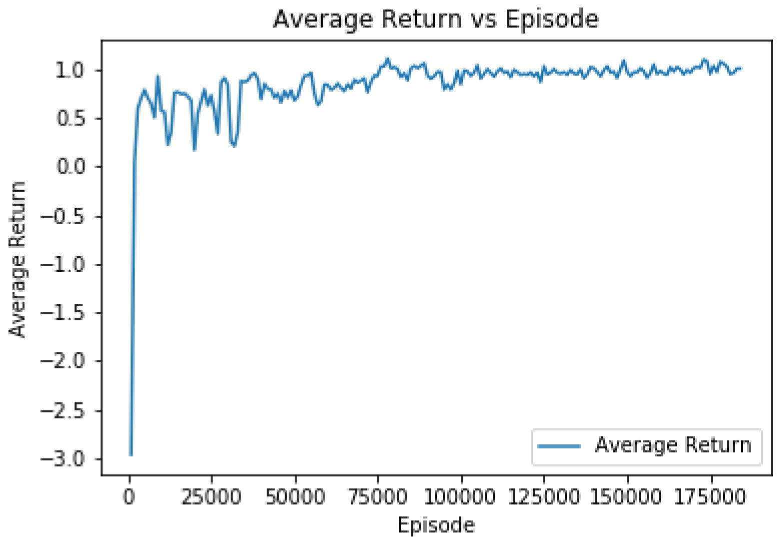

Figure 1: Training curve of model

Over the course of playing 100 games with our trained agent starting first against a random policy, the trained agent achieved the following win/tie/lose rate:

Win Rate: 85.0%
Tie Rate: 4.0%
Lose Rate: 11.0%

For comparison, here are the win/tie/lose rate from project 4, in which our agent always goes first during training. We will refer to this agent as the old trained agent for the rest of the report:

```
Win Rate: 83.0%
Tie Rate: 3.0%
Lose Rate: 14.000000000000002%
```

Hence, we conclude that allowing our machine learning agent to go first only 50% of the time during training does improve its overall performance however marginally. When our trained agent started second against a random policy in over 100 games, its win/tie/lose rate are as follows:

```
Win Rate: 51.0%
Tie Rate: 6.0%
Lose Rate: 43.0%
```

While not remarkable, it is worth to note that a random agent starting second will always lose around 40% of the time. Hence, this marks some form of improvement over the use of a random agent.

And shown below are the charts depicting the win/tie/lose rates across episodes when the agent starts first (top chart) and second (bottom chart). The win rates stayed relatively consistent in the 60-80% range when the agent started first and in the 40-50% range when the agent started second.

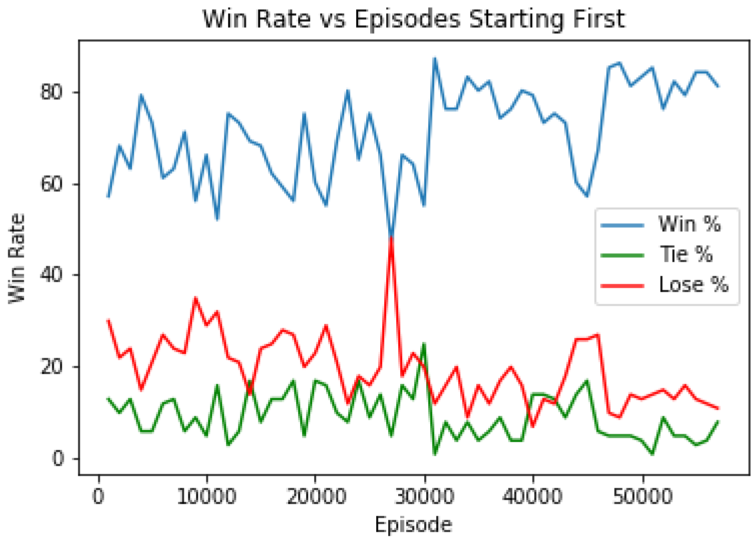

Figure 2: Win/tie/lose rate across episodes when agent starts first

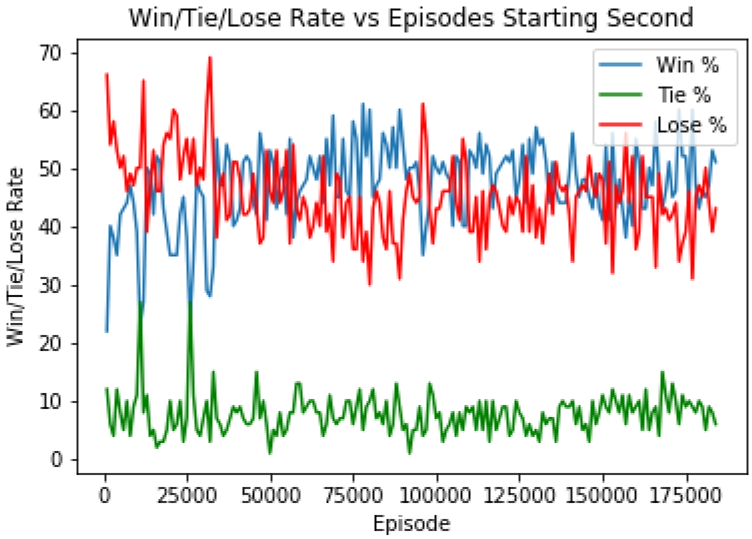

Figure 3: Win/tie/lose rate across episodes when agent starts second

Let us now analyze replays of 5 games to see how our agent gameplay decisions. The first 3 (Game 1, 2, 3) are games in which our agent started first and the next 2 (Game 4, 5) were games where our agent started second. The positions of the game board are as follows:

```
012
345
678
```

Shown below was a game that our trained agent ('x') played while starting first against a random policy ('o'). During turn 3, it is apparent that our agent missed out on a potential game winning play on position 7, instead choosing to make a play on position 0. In project 4, the old trained agent committed similar mistakes as well. We hope that training our agent against itself in part 2 will help reduce these behaviours, since our agent will then be exposed to opponents capable of maintaing some form coherent strategies. Nonetheless, our agent went on to win the game in the next turn when it made the correct play.

```
Game 1, Turn: 1
...
.x.
...
====

...
.x.
o..
====

Game 1, Turn: 2
.x.
.x.
o..
====

.xo
.x.
o..
====

Game 1, Turn: 3
xxo
.x.
o..
====

xxo
.x.
o.o
====

Game 1, Turn: 4
```

```
xxo
.x.
oxo
====

Game 1 result: win
```

In the next game that our agent ('x') played, it is interesting to note that our agent recognized that its path to victory (pos 1-4-7) is blocked and immediately changed its strategy (pos 2-4-6) in turn 3 and 4. In addition, our agent blocked the opponent's ('o') chances of winning when it played position 2. Although this could have happened by chance, we hope that is due to the agent recognizing that this is a 'high reward' move. Likewise with game 1, our agent seems to prefer making plays along the middle column of the board.

```
Game 2, Turn: 1
...
.x.
...
====

o..
.x.
...
====

Game 2, Turn: 2
o..
.x.
.x.
====

oo.
.x.
.x.
====

Game 2, Turn: 3
oox
.x.
.x.
====

oox
ox.
.x.
====

Game 2, Turn: 4
oox
ox.
```

```
xx.
====

Game 2 result: win
```

Game 3 is a great example of our agent's biggest flaws. In turn 4, our agent ('x') chose to make a play in position 8 rather than to block the opponent ('o') move with a play at position 3. Likewise, the old trained agent committed the same mistake in project 4 as well. Once again, our agent's preferred opening strategy is to make plays along the middle column of the board.

```
Game 3, Turn: 1
.x.
...
...
====

.x.
...
o..
====

Game 3, Turn: 2
.x.
.x.
o..
====

.x.
.x.
oo.
====

Game 3, Turn: 3
.xx
.x.
oo.
====

oxx
.x.
oo.
====

Game 3, Turn: 4
oxx
.x.
oox
====
oxx
ox.
oox
====

Game 3 result: lose
```

In game 4, our agent ('x') started second and went for a quick victory. It is apparent by now that our trained agent preferred winning strategy is making plays along the middle column of the game board.

```
Game 4, Turn: 1
...
o..
...
====

...
ox.
...
====

Game 4, Turn: 2
..o
ox.
...
====

..o
ox.
.x.
====

Game 4, Turn: 3
..o
ox.
.xo
====

.xo
ox.
.xo
====

Game 4 result: win
```

Once again, our agent (x') opened with a strategy of going after the middle column of the board. When its path to victory was blocked in turn 3, it immediately changed strategy and went for the top row of the board, achieving a victory.

```
Game 5, Turn: 1
...
o..
...
====

...
ox.
...
====

Game 5, Turn: 2
...
ox.
..o
====

.x.
ox.
..o
====

Game 5, Turn: 3
.x.
ox.
.oo
====

.xx
ox.
.oo
====

Game 5, Turn: 4
.xx
oxo
.oo
====

xxx
oxo
.oo
====

Game 5 result: win
```

From the replay of these 5 games, we note several common themes. The first and most apparent them is that the trained agent preferred an opening strategy of making plays along the middle columns. This in in contrast to the old trained agent, which prefered to make plays along the 0-4-8 diagonal initially. Our new agent is still prone to missing immediate winning plays as we seen in game 1 turn 3. This is perhaps partially due to agent's stochastic nature of picking its next action, however we do hope that training the agent against itself will help eliminate this behaviour. In addition, our agent is still failing to make plays to block the opponent from winning, such as in game 3.

Problem 2

Part 2*: Self-Play*

The next logical step in improving our trained agent is to implement self-play, in which our agent will play against a version of itself instead of a random policy. The original goal was to start with the best weights from part 1, however,we did not have much success with this approach. In addition, we experimented with Part 1 weights from other episodes to no avail. We hypothesized that this could be due to the trained Part 1 agent preference for a specific stratgey (going for the middle column), hence our Part 2 agent could have possibly overffited when training against the Part 1 agent. Thus our proposed model weill be a self-playing model that started out with no pre-trained weights. The agent would be playing against its current version at any point of time during training. In addition, to add variability to the training process, we let our model train against a random agent occassionally so as to reduce potential overfitting. The final model will play against its current version of itself 25% of the training time while starting first, and another 25% of the time while going second. Furthermore, it will also play against a random agent while starting first 25% of the time and starting second for the remaining 25% of the training process. The training curve of the model is attached below, and the red line depicts the average reward it would have gained when going against a random opponent for the entirety of the training process. Lastly, the same reward structure from part 1 was used in training our agent.

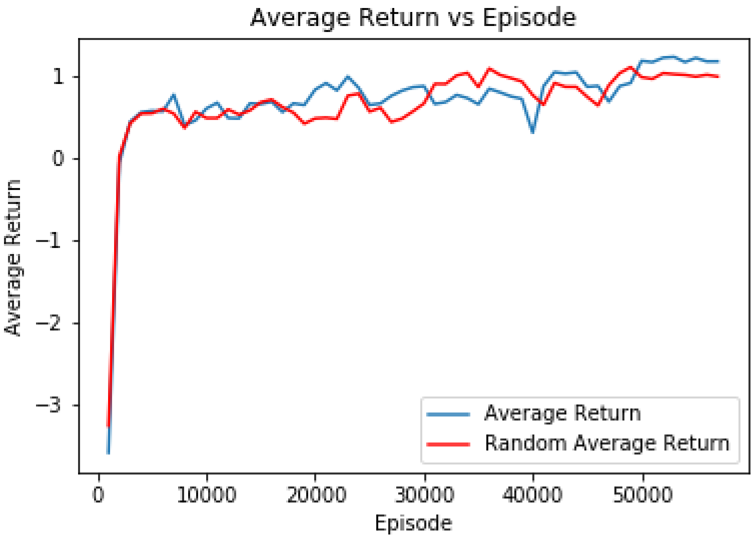

Figure 4: Training curve of model with self-play

Using weights from the final episode (57,000), our trained agent achieved the following results:

Starting First
Win Rate: 83.0%
Tie Rate: 4.0%
Lose Rate: 13.0%

Starting Second
Win Rate: 63.0%
Tie Rate: 3.0%
Lose Rate: 34.0%

The biggest improvement came in the win rate when the trained agent started second against a random opponent, achieivng a win rate of 63%. The win rate against a random policy when starting first is a respectable 83%. These are the win/tie/lose rates against a random opponent across the episodes.

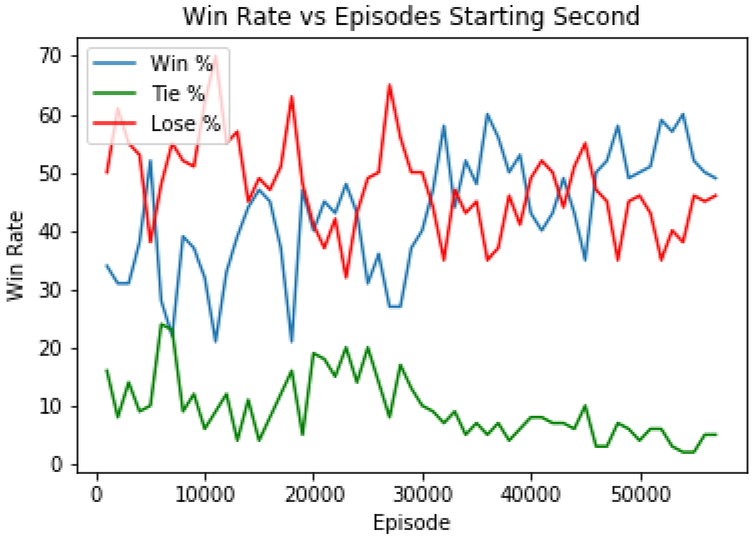

Figure 5: Win/tie/lose rate across episodes when agent starts first

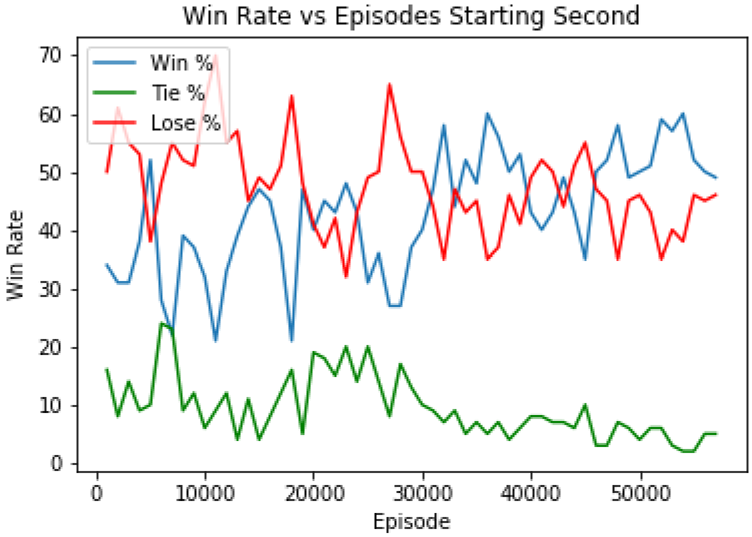

Figure 6: Win/tie/lose rate across episodes when agent starts second

Let us now analyze the performance of the agent in 5 games against a random policy to observe how much self-play has improved the performance of our agent. Once again, the first 3 (Game 1, 2, 3) were games in which our agent started first and the next 2 (Game 4, 5) were games where our agent started second. The positions of the game board are as follows:

```
012
345
678
```

In game 1, our trained agent ('x') once again missed out on a potential game winning play (position 0) in turn 4, instead making a play on position 7. Similar obervations were noted from part 1. Nonetheless, our trained agent won the game in the next turn. It is interesting to note that the trained agent preferred to start on the diagonals this time after training with self-play. In addition, the trained agent seemed to place a great emphasis on playing a 'fork' (positions 4, 7 and 8) that will allow it to win in two ways. Similar strategy was also seen in part 1 (Games 1, 3 and 5), however it seemed that the agent employed the strategy earlier and with greater effect in part 2.

```
Game 1, Turn: 1
...
...
..x
====

...
..o
..x
====

Game 1, Turn: 2
...
.xo
..x
====

.o.
.xo
..x
====

Game 1, Turn: 3
.o.
.xo
.xx
====

.o.
.xo
oxx
====
Game 1, Turn: 4
xo.
.xo
oxx
====

Game 1 result: win
```

In game 2, our trained agent ('x') once again failed to make the winning play (position 6) in turn 3, instead making a play on position 2. Coincidentally, this play blocked a potential winning path for the opponent ('o'). This may have been a learned behaviour, however, it is worthy to note that our agent did not block another winning play by the opponent in turn 4, and hence losing the game.

```
Start of Game 2
Game 2, Turn: 1
...
...
..x
====

o..
...
..x
====

Game 2, Turn: 2
o..
...
.xx
====

oo.
...
.xx
====

Game 2, Turn: 3
oox
...
.xx
====

oox
o..
.xx
====

Game 2, Turn: 4
```

```
oox
ox.
.xx
====

oox
ox.
oxx
====

Game 2 result: lose
```

In game 3, we see a relatively quick victory by our agent ('x') in which it made a winning play through the middle column in 3 turns. A similar strategy was noted in part 1.

```
Start of Game 3
Game 3, Turn: 1
...
.x.
...
====

o..
.x.
...
====

Game 3, Turn: 2
o..
.x.
.x.
====

o..
.x.
ox.
====

Game 3, Turn: 3
ox.
.x.
ox.
====

Game 3 result: win
```

In game 4, our agent ('x') once again focused on creating a 'fork' (position 4, 6 and 7) that allowed it to win during turn 3 in two ways (position 1, 2). However, we do note that the agent once again failed to recognise that it could win the game in turn 3 by just playing position 1. Nonetheless, our agent still went on to win the game eventually by completing the 'fork'.

```
Game 4, Turn: 1
...
...
..o
====

...
.x.
..o
====

Game 4, Turn: 2 o..
.x.
..o
====

o..
.x.
.xo
====

Game 4, Turn: 3
o..
ox.
.xo
====

o..
ox.
xxo
====

Game 4, Turn: 4
o..
oxo
xxo
====

o.x
oxo
xxo
====

Game 4 result: win
```

Game 5 saw the return of the mddle column strategy which allowed our trained agent ('x') to win the game in 3 turns.

```
Game 5, Turn: 1
...
...
..o
====

...
.xo
...
====

Game 5, Turn: 2 ...
.xo
..o
====

...
.xo
.xo
====

Game 5, Turn: 3
...
.xo
oxo
====

.x.
.xo
oxo
====

Game 5 result: win
```

From the replay of these 5 games against a random agent, we saw the constant reoccurence of two strategies employed by our agent, first of which was to go for the middle column of the playing board, the other was to create a 'fork' that could allow it to win in two ways. It was also interesting to note that the agent displayed greater variability in its starting strategy, and it displayed a nice balance between starting in the corner edges or the middle. This greater versatility might have occured as a result of self-play. In project 4, our old agent mostly showed a preference for starts on corner edges, and in part 1 of the bonus, we saw the agent mostly preferring a 'middle' start.

The next replays of 5 games will be against a trained opponent, in which our agent started first.
In game 1, our trained agent won a quick victory by playing the diagonal (position 0,4,8) in 3 turns. It is interesting to note that the trained opponent was not attempting to block our agent's move.

```
Game 1, Turn: 1
...
```

```
...
..x
====

...
...
.ox
====

Game 1, Turn: 2
...
.x.
.ox
====

.o.
.x.
.ox
====

Game 1, Turn: 3
xo.
.x.
.ox
====

Game 1 result: win
```

Game 2 saw another quick victory by our agent ('x') as it went straight for the bottom row (positions 6, 7 and 8) in 3 turns. Surprisingly, our agent seems to be recognising winning plays much better when it is playing against a trained opponent. Once again, the trained opponent seemed to prefer playing positions 1, 4 and 7 in the first 2 games.

```
Game 2, Turn: 1
...
...
..x
====

...
.o.
..x
====

Game 2, Turn: 2
...
.o.
.xx
====
.o.
.o.
.xx
====

Game 2, Turn: 3
.o.
.o.
xxx
====

Game 2 result: win
```

Game 3 was an exact replica of game 2. Our agent won the game easily in 3 turns by employing the same strategy. Once again, the trained opponent seemed restricted in its gameplay approach, preferring to play the middle column even when there isn't a clear victory path.

```
Game 3, Turn: 1
...
...
..x
====

...
.o.
..x
====

Game 3, Turn: 2
...
.o.
.xx
====

.o.
.o.
.xx
====

Game 3, Turn: 3
.o.
.o.
xxx
====

Game 3 result: win
```

Game 4 saw our agent performing a 'fork' to achieve a quick victory against its trained opponent. Our agent however failed to recognize a game winning play in turn 3 where it could have played position 2 to win the game.

```
Game 4, Turn: 1
...
.x.
...
====
...
.x.
..o
====

Game 4, Turn: 2
...
.x.
.xo
====

.o.
.x.
.xo
====

Game 4, Turn: 3
.o.
.x.
xxo
====

oo.
.x.
xxo
====

Game 4, Turn: 4
oox
.x.
xxo
====

Game 4 result: win
```

Game 5 saw another repeat of games 2 and 3, in which our agent won relatively quickly.

```
Game 5, Turn: 1
...
...
..x
```

```
====

...
.o.
..x
====

Game 5, Turn: 2
...
.o.
.xx
====

.o.
.o.
.xx
====

Game 5, Turn: 3
.o.
.o.
xxx
====

Game 5 result: win
```

The 5 games against a trained opponent were interesting because they showed our trained agent winning quickly in many of the games. In fact, a vast majority of game replays against a trained opponent were similar. Perhaps this is a reason why our agent seemed to perform poorer against a random oppoenent when it was trained solely against itself, since it had overfiited to a particular strategy. A better implementation of self play, such as updating the policy based on the rewards from both trained agent, would perhaps address this problem.