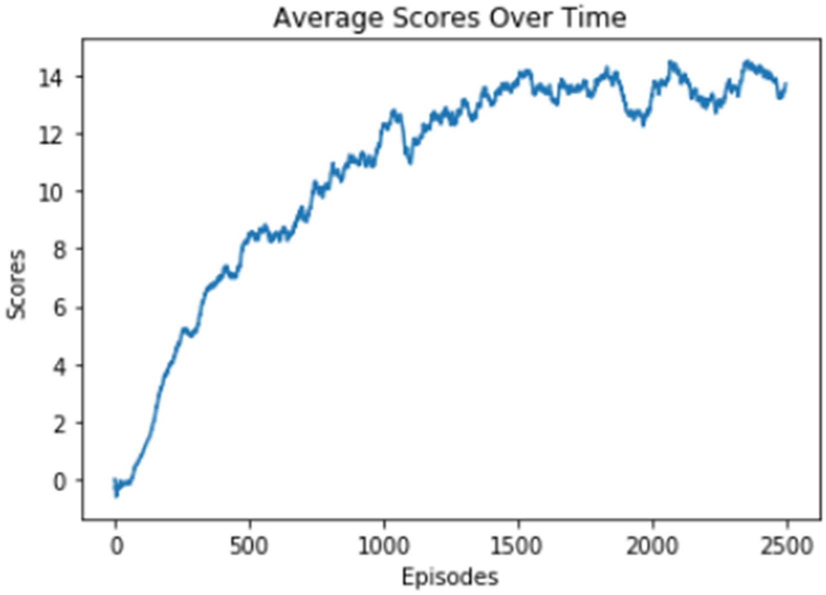
Agent Report

```
Max Average Score: 14.52
```



ACHIEVEMENT:
The Agent received a Max Average Score (over 100 Episodes) of 14.52 at ~2,100 episodes. The number of episodes needed to solve the environment and achieve a 13+ average score was ~1,300 Episodes.

HYPER-PARAMETERS:

```
BUFFER_SIZE = 10000
BATCH_SIZE = 256
GAMMA = 0.95
LR = 1e-4
HIDDEN_SIZE = 128
UPDATE_EVERY = 4
EPISODES = 2500
eps_start = 1
eps_decay = 0.995
```

To start, the BATCH_SIZE of 256 was chosen to give the model a broad training set to train with and be consistent. GAMMA was set at 0.95 to maintain most of the reward threw the actions. A value greater than the value for GAMMA would most likely help the agent. A Leaning Rate of 1e-4 was chosen since it preformed the best amongst 1e-2 and 1e-3. Last, UPDATE_EVERY was set to 4 instead of 50 since it greatly outperformed the former.

LEARNING ALGORITHM:

$$Targets = rewards + GAMMA * QNextStateMax * (1 - dones)$$

The Algorithm that was chosen was SarsaMax (_SarsaQUpdate()) with only a single model. There was an addition to the update rule "(1-dones)" to make sure only the current rewards were included in a terminal state. To calculate the 'Targets,' the sum of rewards for that state was added to the future state Q-Value from the network, times the decay rate GAMMA, times a value of one if not done or zero if at a terminal state.

MODEL ARCHITECTURES:

The model that was chosen was a two layer fully connected Neural Network with ELU Activation. The unit size was as follows, 37 input units, 128 hidden units that worked well for the model, and four action outputs. A Single layer Network, and a three layer network did not achieve the performance of a two layer network, and hence why the two layer Network was chosen.

IDEAS FOR FUTURE WORK:

My first idea would be to create Prioritized Experience Replay since the reward signal is most often sparse. My Prioritized Experience Replay would focus on the magnitude of the reward signal with the highest priority. By using this method it would most likely Increase the Agent to have greater scores. Last, I would like to see how different Learning Algorithms preform in this scenario. Specifically, SarsaMax, Expected Sarsa, and Double DQN, and graph there results over specific parameter range. This would include three values for Gamma(0.93, 0.95, 0.99) and Epsilon Decay(0.99, 0.995, 0.998) for each Learning Algorithm. Then, aggregate the graphs to see which algorithm preformed the best, why that was so, and include that algorithm to improve the agent's performance.