Report:

Network Architectures:

I did not change the network architecture a lot and decided to work with a rather vanilla implementation of [input size, 128, 128, output size] layers structure with Relu activation in the hidden layers and batch normalization. The reason for not paying too much attention to the network was because finding the optimal network architecture is a problem with a complex high dimenstional space and would take many hours of training to find the right one, while many times finding a bad one. Since this is not the purpose of this excercise, I preferred focusing my attention on other aspects of the project

Further improvements:

An improvement that was already suggested in the benchmark implementation in the project description was gradient clipping:
I used gradient clipping as suggested in the project description - the effect was pretty minor

```
torch.nn.utils.clip_grad_norm(self.critic_local.parameters(), 1)
```

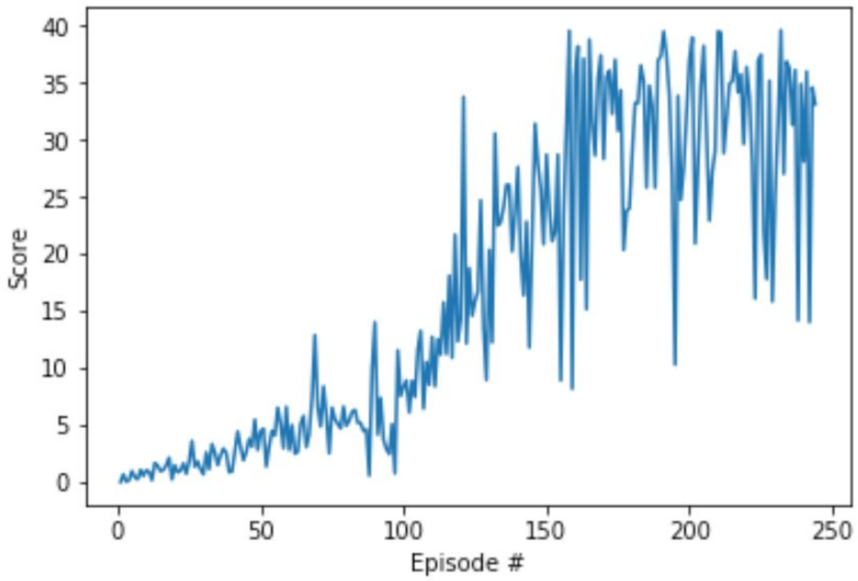
An idea I got from a Slack discussion was to set the weights of local and target actor, respectively, local and target critic to the same values:

```
self.hard_copy_weights(self.actor_target, self.actor_local)
```

Results:

The training starts with rather modest linear improvement until episode 100, with an average score of 3.55, episodes 100-200 produce a huge jump in performance to 23.62 average score. The environment is solved after 244 episodes with the target of 30 average score over 100 episodes

```
Episode 100     Average Score: 3.55
Episode 200     Average Score: 23.62
Episode 244     Average Score: 30.01
Environment solved in 244 episodes!     Average Score: 30.01
```



Further work:

While the environment is solved, we can see that even in the last 100 episodes, there are occasional lapses in performance and the training is not as stable as we'd like it to be. In order to tackle this, we can focus on better sampling from the experience replay, epsilon decaying greedy policy and general optimization of the NN models with dropout, more layers, bigger layers etc