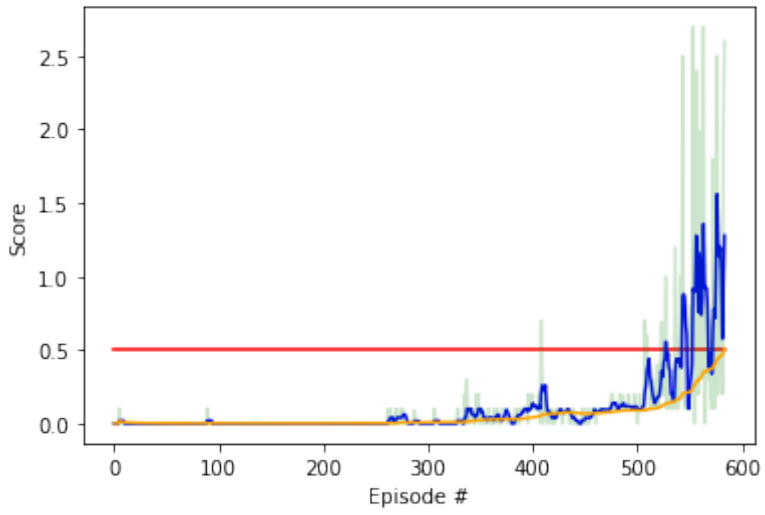
Report

In my implementation I used the code for a MADDPG-agent to train two agents to play tennis with each other while they use the same actor and the same critic network.
The jupyter-notebook „Tennis.ipynb“is the main file, where we load the environment, set up the agent and are able to monitor the progress. When the agent reaches an average reward of 0.5 over the 100 episodes, it stops, saves the model and returns a plot of the learning curve of the agent.
The agent itself uses an actor network to compute the next action. We used tanh-activations at the end to get numbers between -1 and +1. To let the agent explore the environment, we always sample a small error from a normal distribtuion with $\mu = 0$ and $\sigma = dir0o$ which we multiply with ϵ before we add it to the action. We start with an $\epsilon = 1.$ and use an exponential decay with decay rate 0.95. We use a lower bound for ϵ: $\epsilon_{min} = 0.05$.
Additionally, we constructed a critic to rank the effect of the action and compute the Q-values. The agent has a replay puffer of size 10000, where it stores the different experiences it makes. We update the networks after each step. The other hyperparameters are:

$$\gamma = 0.99\,, \qquad \tau = 0.01\,.$$

The actor takes as input the current state. It has 3 dense-layer, 2 layers with 256 neurons, and the output-layer has 2 neurons. The hidden-layers have ReLu-activations, while the output layer has a tanh-activation which is trained using Adam optimizer with a learning rate of 10^{-4} and a batch size of 128. The critic takes as input the state and the action of the step. It has 3 dense-layer, 2 layers with 256 neurons, and the output-layer has 1 neurons. The hidden-layers have ReLu-activations. We train the critic network with a learning rate of 10^{-3} with Adam optimizer and a batch size of 128.
As a result we get for the average reward the following plot:



As we can see is our agent able to solve the environment within 0 steps. The orange line is the average line over the last 100 episodes, the green line is the actual reward for every episode, while the blue line gives an average over 5 consecutive episodes which gives a better impression of the actual learning progress.

For the future,we can try to simplify the network. Another possibility is to improve the hyper-parameters further. I will also test this algorithm on the environment „soccer“to see whether it generalizes to more complex environments.