**Assignment 2- policy gradients**

**Niv Baron: 204351944**

**Elad Feldman: 308385947**

**Section I**

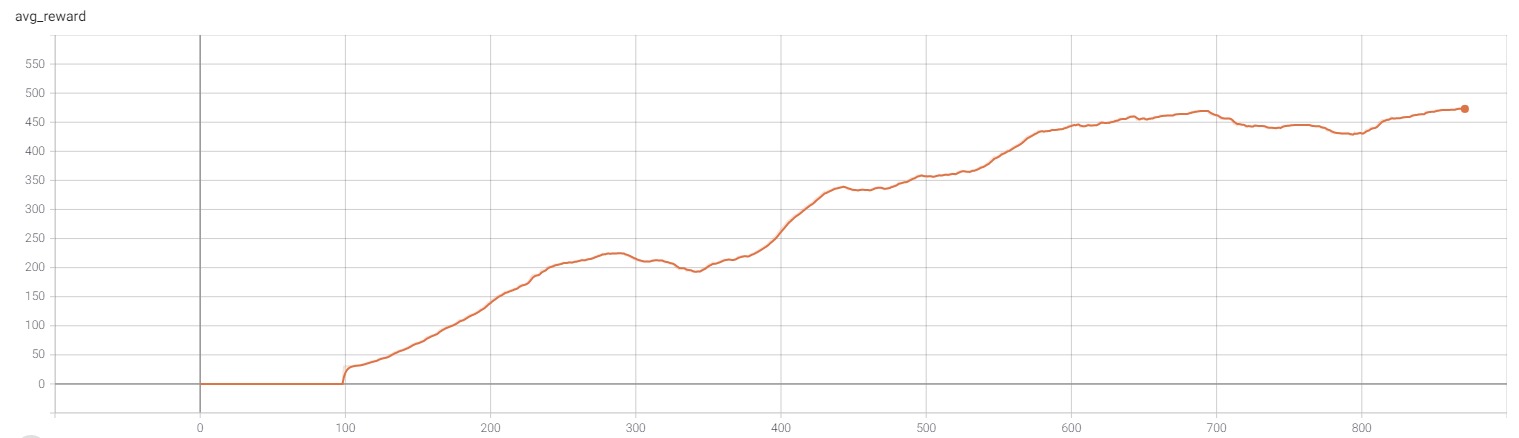
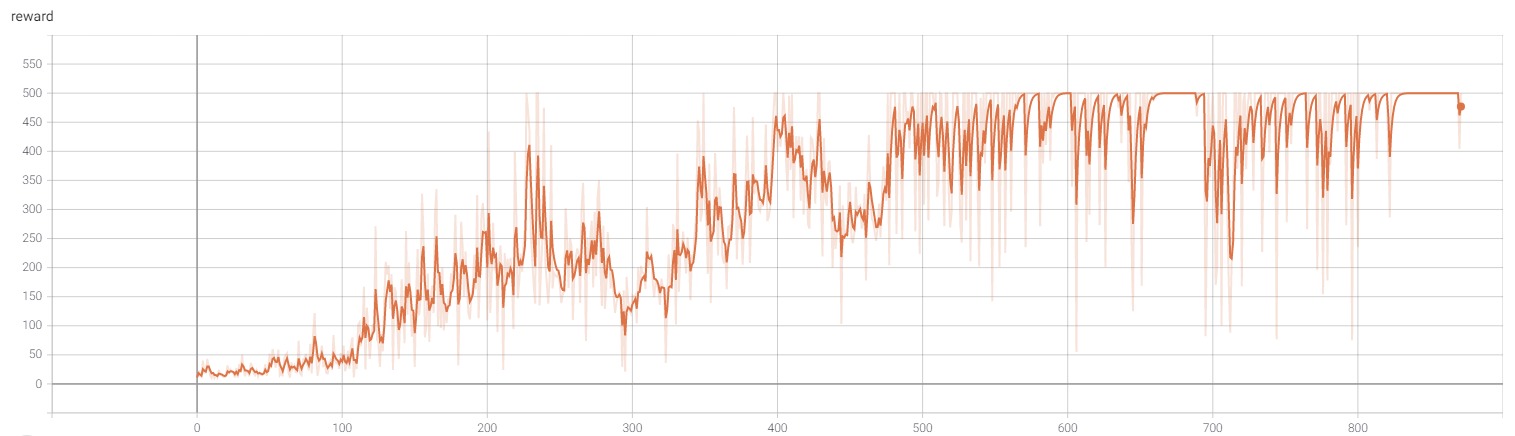
The value of the advantage estimate reflects the contribution of the action beyond the current “average” of the state. Because the return can be change drastically between episodes, the REINFORCE algorithm gradient can suffer from high variance. This variance leads to unstable convergence. By subtracting the baseline, we can reduce the variability of the return and therefore the variance of the gradients.

1. The exaptation of the baseline during all episode transition according to policy can be express as the exaptation of the exaptation on each transition.
2. If (and only if) the baseline of state is independent with we can extract it from the expectation due to expectation linearity.
3. Expectation definition.
4. Log derivative.
5. According to Leibniz integral rule the gradient (by ) and the integral (by a) can be interchange.
6. the integral of a density function between and .
7. Gradient of constant equals to zero.

As mentioned in b prerequisite condition for this equation to be valid is that is independent with .

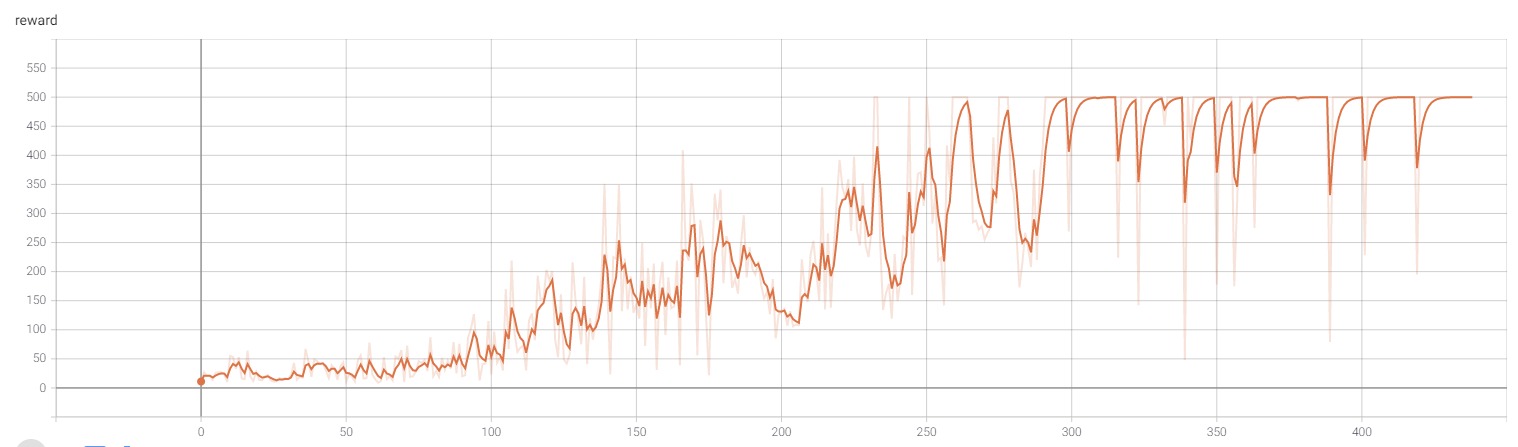
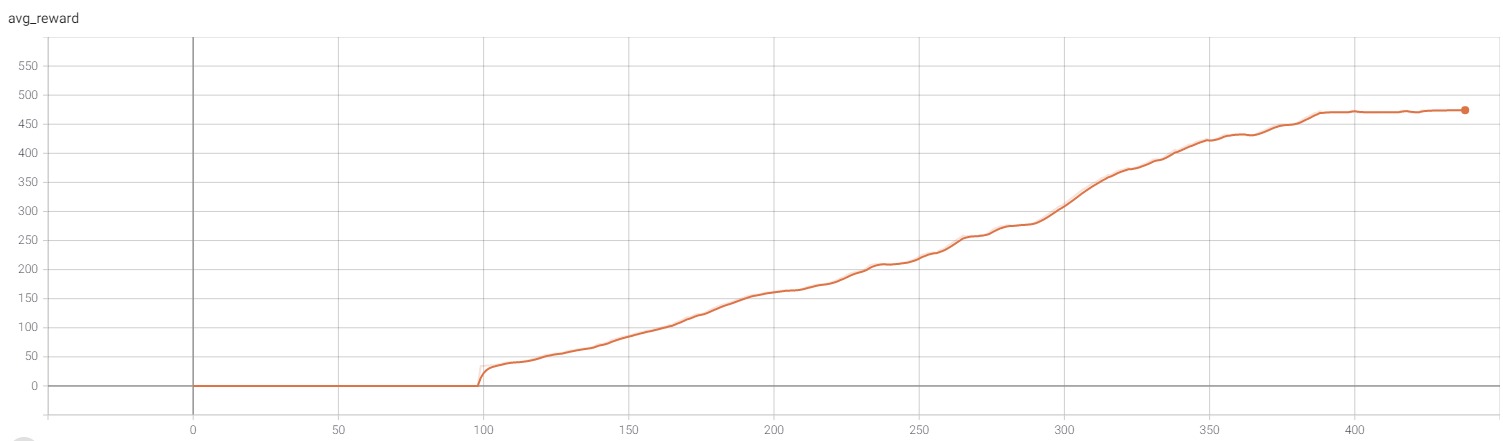
**Programing assignment**

For the programing assignment in this section, we added tensorboard logging to ‘policy\_gradients.py’ script we received and implement REINFORCE with base line in ‘reinforce\_with\_baseline.py’ script, you can run these scripts without any additional parameters to get the results.



**Figure 1- Naive REINFORCE Algorithm results.**

As we can see in figures 1-2 subtracting the baseline helps the REINFORCE Algorithm to converge much quicker (483 episodes vs 871 episodes) it is also demonstrated in these figures that subtracting the baseline helps to reduce the rewards variability which implies a smoother convergence.



**Figure 2- REINFORCE with baseline results.**

**Section II**

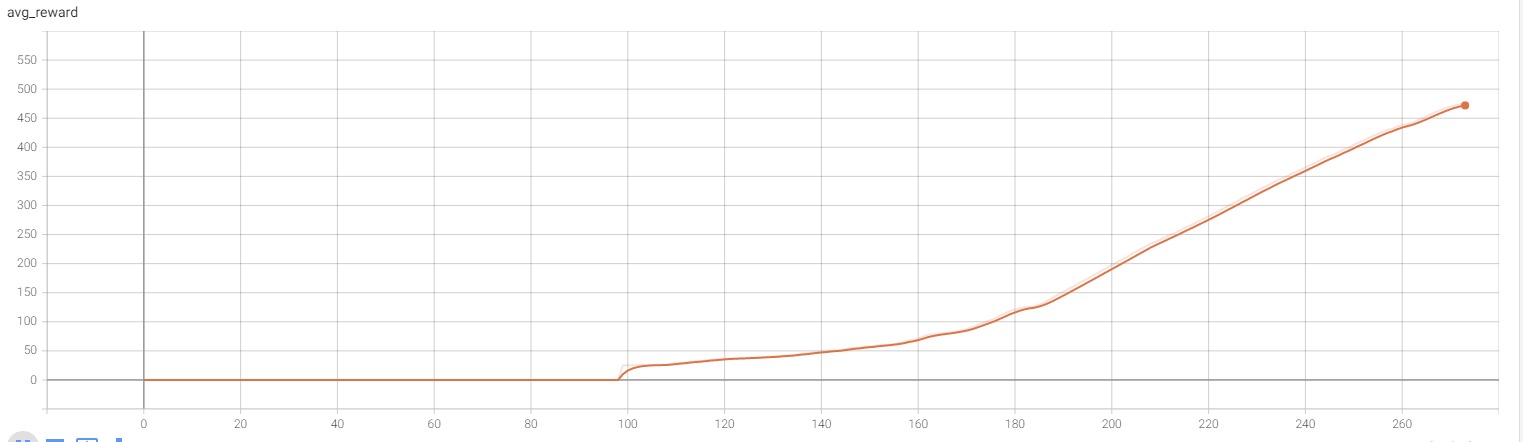
The policy function is playing as the actor and the value function is the critic. The policy is determining the actions when given the state, hence, its role is the actor while the value gives feedback for the taken action and play as the critic.

**Programing assignment**

Here we implemented an actor-critic inference to policy gradience as described in the given algorithm, you can run our implementation by running ‘actor\_critic.py’ without additional parameters.

**Actor-Critic results**

**Figure 3- actor-critic algorithm results.**



As we observed empirically from the results of the different algorithms, the actor-critic inference for policy gradients produced the best results (converged in 273 episodes). We can see in Figure 3 that the ability to optimize the networks during the episode and the value function estimation results in much fester and smoother converges then REINFORCE algorithm (871 episodes) and even REINFORCE with baseline (483 episodes).