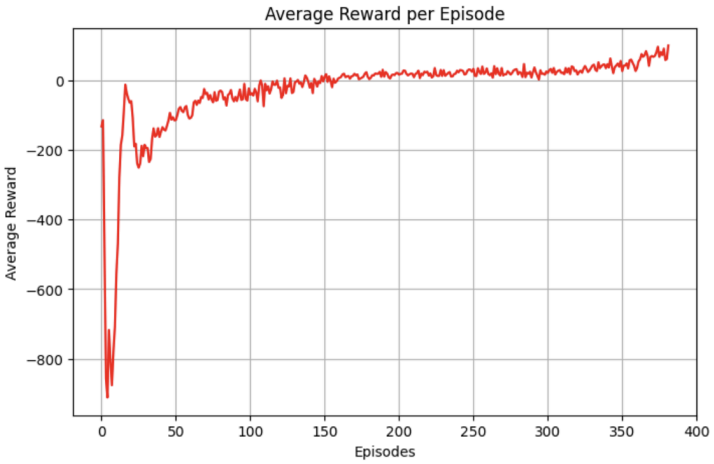
```
Episode 454     Average Score: 107.37
Total duration: 1661.18 seconds
Average time for training: 2.40 seconds
Average time for updating: 0.03 seconds
Average time for synchronizing: 0.04 seconds
```

Performance with default parameters (num_agents = 4, batch_size = 64, num_epochs_actor_train = 1)

```
Episode 381     Average Score: 100.07
Total duration: 1393.49 seconds
Average time for training: 2.50 seconds
Average time for updating: 0.02 seconds
Average time for synchronizing: 0.03 seconds
```



Performance with batch_size = 128

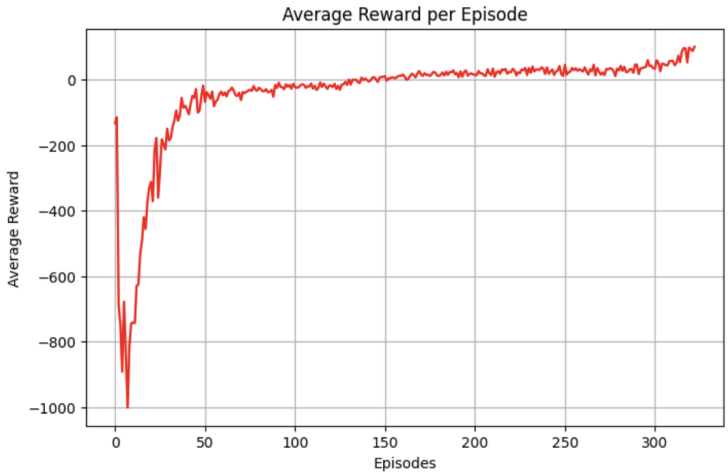
Increasing the batch_size parameter from 64 to 128 significantly improved the quality of the training, with the model now reaching its threshold value 5 minutes faster than it did under default conditions. This makes sense, as the increased number of samples in training allows for a more precise - and therefore smoother - gradient descent, which is reflected in the respective timeline graphs. However, it is worth noting that, due to the increased sample size taking longer per step, the average time for training does slightly increase.

```
Episode 156     Average Score: 101.73
Total duration: 1786.30 seconds
Average time for training: 10.21 seconds
Average time for updating: 0.03 seconds
Average time for synchronizing: 0.03 seconds
```

Performance with num_epochs_actor_train = 4

Increasing the num_epochs_actor_train parameter from 1 to 4 drastically improved the efficiency of the training, with the model reaching its threshold value in 156 episodes compared to 454 episodes under default conditions. However, the average training time nearly quadrupled, and as a result this model actually ended up taking longer to reach the threshold than the default model did. The increased efficiency is due to the agent exploring the samples from each episode in much greater depth, allowing for a faster convergence, while the quadrupling of training time per episode is simply the result of each actor training for 4 times as many epochs per episode.

```
Episode 322     Average Score: 100.97
Total duration: 1595.00 seconds
Average time for training: 3.73 seconds
Average time for updating: 0.04 seconds
Average time for synchronizing: 0.06 seconds
```



Performance with num_agents = 9

Increasing the num_agents parameter from 4 to 9 improves the efficiency of the training (322 episodes to reach threshold as opposed to 454 under default conditions). This is because

the more distributed spread of agents allows for a faster collection of samples, which in turn allows the model to converge faster. This more distributed spread is also reflected in the improved stability of the learning, as can be seen in the timeline graph. However, the reduced computational power allotted to each agent (1 CPU instead of 2) means that each individual agent was slower to train, which is why the average training time per episode is significantly increased. Additionally, having more agents makes the processes of updating and synchronizing all agents longer and more complex, and as such the average updating and synchronizing time per episode increased as well.

Overall, I think this assignment was very helpful for my understanding of how (through Ray, but also as a general principle) processing-intensive jobs can be made more efficient through distributed clusters of actors. Part 2 in particular, by getting to work with a concrete example of MapReduce as a step-by-step process and seeing its efficiency in action, really clarified for me how it works and why it is so beneficial.