ECE 4984: Advanced Robot Motion Planning (Fall 2018)

Homework 3

Due: Monday October 22, 9PM

Corrected on Monday October 15

October 15, 2018

Instructions. This homework constitutes 10% of the course grade. You must work on it individually. It is okay to discuss the problems with other students in class. However, your submission must be your original work. This implies:

- If you discuss the problems with other students, you should write down their names in the pdf report.
- If you refer to any other source (textbook, websites, etc.), please cite them in the report at the relevant places.
- The answers in the pdf report must be written entirely by you from scratch. No verbatim copy-paste allowed without citations.
- Any software you submit must be written entirely by you with no copy-pasting of significant portions of code from other sources.

Please follow the submission instructions posted on canvas exactly. You must submit your assignment online on canvas by the due date. Your submission must include one pdf file with the answers to all the problems and one or more files containing your code for Problem 4. It is okay to scan your answers and create the pdf submission.

Problem 1 — 50 + 20 bonus points

Suppose you have a robot that is operating on the slope of a hill. At any given time, the robot can either be on the top of the hill, in the middle of the slope, or on the bottom of the hill. The robot can turn on its motors to try and stay on the top of the hill.

Every time the robot turns on its motors, it loses one unit of energy. The robot can harvest solar energy depending on its position after executing the action. We have the following possibilities:

- If the robot is at the top of the hill and it turns on its motors for one unit of time, in the next time step it will stay at the top with 0.7 probability and harvest 4 units of energy or slide down to the middle with 0.3 probability and harvest 1 unit of energy. If the robot is at the top of the hill and does not turn on its motors, in the next time step it will stay at the top with 0.5 probability and harvest 4 units of energy and slide to the middle with 0.5 probability and harvest 1 unit of energy.
- If the robot is in the middle and it turns on its motors for one unit of time, in the next time step it will reach the top with 0.4 probability and harvest 4 units of energy, stay in the middle with 0.4 probability and harvest 1 unit of energy, and slide down to the bottom with 0.2 probability and not harvest any energy. If the robot is in the middle and does not turn on its motors, in the next time step it will stay in the middle with 0.5 probability and harvest 1 unit of energy and slide to the bottom with 0.5 probability and not harvest any energy.

- If the robot is at the bottom of the hill and it turns on its motors for one unit of time, in the next time step it will stay at the bottom with 0.3 probability and not harvest any energy and climb to the middle with 0.7 probability and harvest 1 unit of energy. If the robot is at the bottom of the hill and does not turn on its motors, it will stay at the bottom with probability 1 and not harvest any energy.

If the robots ends up with 10 or more units of energy, then we say the robots wins the game. Find a policy to maximize the probability of winning the game.

The amount of energy the robot has is equal to the amount harvested due to solar energy minus the amount lost due to turning on its motors. If needed, you can assume that the robot initially starts with 5 units of energy and in the middle (but you shouldn't really need this assumption).

What to submit?

1.A **(50 points)** Formulate this problem as a Markov Decision Process. Your answer (in the pdf report) must clearly specify the set of states S, the set of actions A, the transition probability $T(s, a, s')$, and the reward function $R(s, a, s')$. You can either list all the aforementioned quantities or explain clearly in words how each term can be calculated. You must also write down the objective function and discount factor.

1.B **(Bonus: 10 points)** Find the optimal policy $\pi(s)$. Your policy (in the pdf report) must state the optimal action for each state s in your state space.

1.C **(Bonus: 10 points)** What is the optimal probability of winning the game when the robot starts with 5 units of energy and in the middle?

In order to find the solution for parts B and C, you can write a program that implements value iteration. This is not strictly required. If you wish, you can compute the optimal solution by hand. In the case you compute the solution by hand, you must show each step of your computation in the pdf report. If you implement value iteration, then you must submit your program (written in C/C++/Python/MATLAB) along with instructions on how to compile the program.

Problem 2 — 20 points

Explain the role of the ϵ parameter in ϵ–greedy policy for action selection in Q-learning. Specifically, what do you expect will happen if you set ϵ to be a very small value and what will happen if you set it to be a large value?

Problem 3 — 30 points

We will use the ConvNetJS MNIST demo for this question. This is an implementation of a neural network that is used to detect which digit the input 24×24 corresponds to.

How to run the code?

- Visit `https://cs.stanford.edu/people/karpathy/convnetjs/demo/mnist.html`
- The sample code should start running. Your screen should look like the Figure 1.
- You will have to edit the code given in the block titled **Instantiate a Network and Trainer**.
- Select the code given in this block and replace it by the code given in problem3.txt.
- Click **change network**.
- The graph will reset (if it does not, it means something has gone wrong).

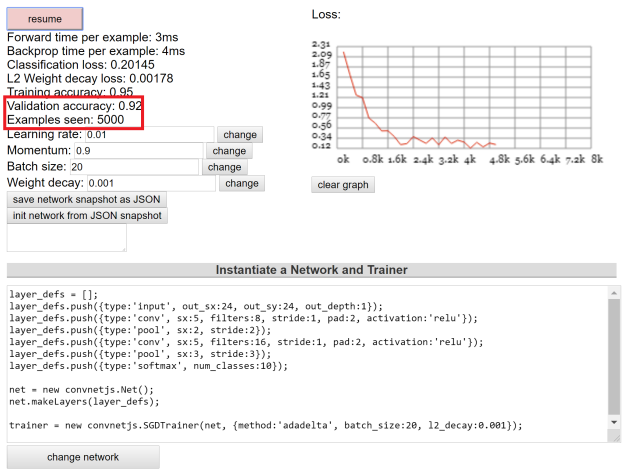

Figure 1: This is how the ConvNetJS sample code should look like in your browser. Note the Validation accuracy and Examples seen fields that you will need to monitor for Problem 3.

What to submit?

3.A **(10 points)** Design a neural network with only one hidden layer. The hidden layer must be a fully-connected (fc) and must use sigmoid activation. Vary the number of neurons in the hidden layer from 5 to 50 in steps of 5. In the pdf report, plot the validation accuracy after running approximately 10,000[1] training examples, as a function of the number of hidden neurons.

3.B **(5 points)** Repeat the same as 3.A using the relu activation function.

3.C **(5 points)** Repeat the same as 3.A with a neural network with two hidden layers with sigmoid activation function. You must vary the number of neurons in both hidden layers. Pick at least 3 values for the number of neurons in the first layer and 3 for the second layer, giving a total of 9 combinations.

3.D **(5 points)** Repeat the same as ~~3.B~~ 3.C using relu activation function.

3.E **(5 points)** Find a neural network with any number of hidden layers and neurons using any combination of relu and sigmoid activation function that gives the highest validation accuracy using approximately 10,000 samples. In the pdf report, you must give a screenshot showing the final accuracy results for your network as well as describe the architecture of your network.

[1]Unfortunately, there is no easy way of stopping the learning after 10,000 examples in this demo. You will have to hit the pause button when the number of examples reaches close to 10,000. Therefore, for this assignment approximately 10,000 training examples will mean any number between 9,500–10,500.