CS 3600 Final exam

Due December 14, at 11:59pm Eastern

Name: Donaven Lobo

List any collaborations here: Rohan Nandakumar, Fernando Sanz

Question 1. In each of the following questions, you will be given a scenario and be asked to consider whether a number of possible algorithms may or may not be well-suited to the task. One sentence should suffice for each. (1 point each)

1.a. A robot must navigate a collapsed mine to bring food and water to some survivors. If it falls down a shaft it will be destroyed. Air breezes are detectable nearby to most shafts, but breezes are sometimes too faint to be detected. Additionally, the rough terrain means that the robot's feet may slip and it may not always move forward at the intended pace. Explain why you should or should not use each of the following to reach the survivors:

i) A*: This is not a viable choice since the map of the mine is not known due to it being collapsed. Therefor this algorithm would not be able to find the shortest path.

ii) A Markov Decision Process: This is a viable choice since it turns it into a sequential problem and determines the optimal action to take at a given state to reach the survivors and avoid the shafts.

iii) A Partially-Observable Markov Decision Process: This is the best choice for the given scenario since the environment of the mine if not fully observable and also there is both action and sensor stochasticity that this model accounts for.

iv) A Dynamic Bayes Network: This model could be used to show the probability distribution of robots' actions and its probability of falling down a shaft based on sensor input. It does not execute any actions itself however.

v) A non-Dynamic Bayes Network: You should not use this model at all since the collapsed mine may be constantly changing which will not be factored in any predictions.

1.b. In the game of curling, the objective is to slide a stone down a sheet of ice and try to get it as close to the center of a target as possible. You release the stone at a particular velocity, however there are small imperfections in the ice. Explain why you should or should not use each of the following to predict where the stone will stop.

i) A*: I would not recommend using A* since although it would find the shortest path to the center, it would not account for other factors such as the imperfections in the ice or the changing game state.

ii) A Markov Decision Process: I think a MDP would be the best choice for this case since we have a fully observable environment, and we can take the best action based on what given state we are in accounting for action stochasticity.

iii) A Partially-Observable Markov Decision Process: I would not use this model since our environment is fully observable and we can make definitive decisions based on that.

iv) A Dynamic Bayes Network: I would not use this model since there are too many factors that could influence the curling stones position and it would be difficult to model with a DBN. There is no obvious cause and effect relationship to model for this game.

v) A non-Dynamic Bayes Network: I would not use this model due to the same reasons as above in addition to the model not considering the dynamic nature of the game.

1.c. Suppose you need to assemble a piece of Ikea furniture but lost the instructions. You need to figure out what order to assemble the pieces in, starting with a pile of parts. You have a very powerful computer that doesn't have the internet. Assume that since you will build the furniture according to the instructions generated, you will not have any problem identifying parts or applying each step. Explain why you should or should not use each of the following to create the sequence of steps to build the furniture:

i) A*: I think this a good algorithm to use for this task since it will attempt to find the fastest way to assemble the piece of furniture and there is sensor and action determinism which allows for all the intended steps to execute as intended.

ii) Reinforcement Learning: This could also be a good choice since there could be reward associated with different checkpoints in the building process. However, it would be difficult to pre-assign this.

iii) A Dynamic Bayes Network: I don't think this model should be used since we know the environment is completely deterministic and so it would not give us any new information.

iv) Simulated annealing: I don't think this model should be used since it is not guaranteed a solution would be found and it may not be the best idea to start at different pieces when assembling furniture.

v) A Perceptron: I don't think this model should be used since it requires a lot of training data and may over complicate the task.

1.d. Suppose you want to classifying rodents found on campus by species. There are 3 types of rodents (rat, shrew, or, mouse) that can be identified by inspecting size, color, tail length, whisker length, and size of front teeth. You must consider that any of these attributes can take on a range of values (e.g., a baby rat could be the size of a full-grown mouse). Explain why you should or should not use each of the following:

i) A*: This should not be used since it doesn't require finding an optimal path

ii) A Markov Decision Process: This should not be used since the problem doesn't contain states and actions to make.

iii) A Dynamic Bayes Network: This model could be used if there are suitable sensors that could help create an emission model to identify the probability of each type of rodent. I would still suggest against it.

iv) A Perceptron: I think this is the best choice for this problem since the model can figure out the importance of given features on its own allowing it to learn how to identify the correct rodent. Additionally, it would be trained to identify rodents it never saw before.

v) Simulated annealing: This should not be used since this is used to optimize a certain process/ function however we are trying to identify something.

1.e. Your job is to transport containers overseas. Containers can hold different things and can sometimes be very heavy or very light. Depending on what is in the container, you can charge more or less money no matter if it is heavy or light. A container ship can hold a certain amount of weight. You must put as many containers on a ship as possible without going over the weight limit and that will make as much money per shipment as possible. Explain why you should or should not use each of the following to figure out which combination of containers to put on a ship:

i) A*: This could be used if we state that the goal state is our maximum weight, and the fastest route is the one that holds the most value. However, this assumes we know all the possible containers we have at our disposal prior.

ii) A Bayesian Network: This should not be used because this mainly used for cause-and-effect relationships.

iii) A partially observable Markov Decision Process: This could be used making it a sequential problem based on what container you get next and deciding on how you can charge the most given the amount of space you have left.

iv) A Perceptron: This might be a good option since there may be many factors to consider when deciding whether a container should be taken or not. It could possibly take into account what we got in the past and probability of a high value crate in the future.

v) Simulated annealing: This could also be a good option since it could help find the optimal amount to charge and the optimal size of container to accept allowing for the most money per shipment.

Question 2. (2 points) Suppose an autonomous car is driving down a road that is passing through a forest. The car strikes a pedestrian crossing a street. The pedestrian was dressed in a Halloween costume that made them look like an *Ent* (a walking, talking tree from The Lord of the Rings). It was raining at the time. The autonomous car uses only camera sensors. The car makes decisions using deep reinforcement learning. You have been called in to help investigate the crash.

You remember from CS 3600 that your professor told you there were five potential causes of errors. Explain how each type of error could have caused the crash using specific details from the scene.

For example: if the car was performing online learning and it had never seen an *Ent*, it might have chosen to explore what would happen if it accelerated instead of braking.

(I have completed one for you, give the other four)

There are 5 potential harms: Sensor error, Effector error, Model error, Online Learning, and a Wrong Objective function.

Sensor Error: The camera sensors may have misinterpreted the person in the costume as just some leaves blowing across the road, and so it continued to drive through it.

Effector Error: The camera may have correctly identified the person however it might have been too late to stop due the rain impeding visibility or not allowing the car to get off the path due to reduced traction.

Model Error: The car may have just made the incorrect choice for the state it was in. This could have been due to the model never encountering a similar scenario before and therefor guessed on what it should do. Or it could have been due to bad training data/ the model not reinforcing the correct thing to do in a new scenario.

Wrong Objective Function: It is difficult to give a driverless car a good objective function. It might just be to get the passenger from point A to point B as quickly as possible. However, the objective function may be incomplete and, in this case, maybe the car didn’t care about pedestrian safety and instead was more concerned about still getting the passenger to their destination in a set time.

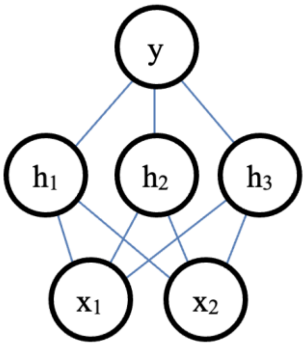
Question 3. (1 point) You are developing a machine learning system for your e-book company to predict whether a person will read a book or not. As a good machine learning engineer you try a variety of machine learning algorithms and a variety of parameterizations of those algorithms. You discover that two algorithms, deep neural networks and decision trees, work best on the data. Your boss wants you to deploy the deep neural networks so that their marketing team can tout the use of a popular technology. What justification could you give to argue for decision trees instead?

First, a decision tree will be much easier to create and model a system with compared to a deep neural network. A decision tree would run much faster at inference since it is way less computationally intensive. On the other hand, a neural network takes a long time to train and requires a lot of training data. It would also be much easier to understand how the decision tree came to a given conclusion compared to an ML model. The neural network would assign weights to factors we may not think are that important and don’t make much sense to us, hence the recommendations may be hard to support sometimes. Finally, some historical data might lead the ML model astray while it is easier to spot inconsistencies in a decision tree and could instead justify the removal of that data.

Question 4. Neural networks. Recall that the Rectified Linear Unit (ReLU) activation function is defined as:

$$g(x) = \begin{cases} x, & x > 0 \\ 0, & x \leq 0 \end{cases}$$

Consider the following neural network:



The activation function for all nodes are ReLUs. Neurons h_1, h_2, h_3, and y have a bias of -1. All weights, including the bias weights, are initialized to 0.1.

4.a. (1 point) x_1 and x_2 are input nodes. Suppose they are given the following values from one data point:

$x_1 = 3.0$
$x_2 = 2.0$

Compute the output activation of node *y*. (show work for partial credit)

output of $y = 0.02$

4.b. (1 point) Suppose the target value should have been **0.0**. Using the loss function

$$L(target, output) = \frac{1}{2}(target - output)^2$$

and the and output of y that you computed from part 4.a, compute w_{x_1,h_1}, which is the weight between nodes x_1 and h_1, after back propagation. Use a learning rate of $\alpha = 0.5$.

$w_{x_1,h_1} = 0.097$

(Hint: you can verify your answer by using updated values for the weights in another forward pass to see if the output of *y* is closer to the target)

4.c. (1 point) Suppose at some point in the future of the neural network training the weights are:

$w_{x_1,h_1} = w_{x_2,h_1} = w_{x_1,h_2} = w_{x_2,h_3} = w_{x_1,h_3} = w_{x_2,h_3} = w_{bias,h_1} = w_{bias,h_2} = w_{bias,h_3} = 0.08$

$w_{h_1,y} = w_{h_2,y} = w_{h_3,y} = w_{bias,y} = 0.09$

Run a forward pass on the neural network using the following data:

$x_1 = 0.6$

$x_2 = 0.6$

Compute the output activation of node y. (show work for partial credit)

output of $y = 0$

4.d. (1 point) Suppose the true target value is 1.0 for the data point in 4.c. Explain what will happen to the weights after the error from the output of y is backpropagated. (Hint: computing the back-propagation step may help though we don't require you to give us the new weights). Why is it different than what happened in 4.b.?

The perceptron weights should increase, and the bias weights should decrease since our prediction was too low. This is different from 4b since we overestimated at that point and therefor needed to decrease the perceptron weights and also increased the bias weights to reduce our overall output.

4.e. (1 point) It's not good for a network to be sensitive to the choice of a bias value. Instead of fiddling with the bias value, your professor, who is a genius,* has invented a new activation function called the Markified Linear Unit (MaLU), which is defined as:

$$g(x) = \begin{cases} x, & x > 0 \\ 0.001x, & x \leq 0 \end{cases}$$

Explain why the Markified Linear Unit will work better on the neural network above.

The MaLU activation function accounts for the bias by not eliminating any negative inputs that come into a node. This allows for negative values to propagate up and eventually cancel out after time. It could be seen as a way of negative feedback that tells the system that there is a negative input coming from somewhere and how large it is.

* Allegedly, it has not been confirmed.

Question 5. The gradient descent used to train neural networks is a form of greedy hill-climbing (although going down instead of up). As such, it can get caught in local minima.

5.a. (1 point) Simulated annealing can get out of local minima. So why don't we use simulated annealing to train neural networks?

While simulated annealing may get you out of a local minima, it doesn’t guarantee an optimal solution. Many times it leads to a sub optimal solution since it doesn’t fully explore areas and can lead to exploring another area before that minima has been found. Also given that our loss function has a derivative, it is much simpler to implement a gradient decent algorithm compared to simulated annealing.

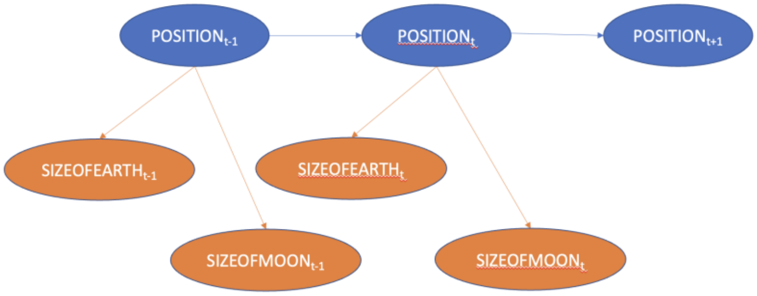
5.b. (1 point) Some more advanced gradient descent algorithms introduce the idea of “momentum” where a fraction of the delta of a weight from the previous iteration of back-propagation is added to the current delta of a weight from the current iteration of back-propagation. How does it help with the problem of local minima?

Having this momentum in gradient algorithms helps break through the local minimas and suboptimal solutions and continue to search for a global minimum. It would go past the local minimum and slow down, but it would hope to continue momentum into another minima. This overcomes the problem of getting stuck at local minimums since it will always carry momentum over the minimum just in case.

5.c. (1 point) Another advanced option for back-propagation is to use a decaying learning rate. The learning rate (alpha) starts relatively high and gets smaller after every epoch. What problem that we discussed in class does a decaying learning rate help with?

A decaying learning rate tackles the problem of the algorithm reaching a point of minimum error and then stops making progress. When the learning rate is too high, we may overshoot the global minimum and when the learning rate is too low, it may take a long time for the algorithm to converge. Having a high learning rate at the start allows for the algorithm to quickly make progress at the start and then slowly approach the global minimum. This allows the algorithm to work much faster as well as avoid getting stuck at local minimas.

Question 6. (1 point) Suppose you are an astronaut on the way to the moon. Unfortunately, your radar has gone out and you have lost contact with Mission Control on Earth. You want to be able to estimate your current position and predict when you will get to the lunar orbital insertion point so you can fire your thrusters and enter orbit. You figure your current position is related to the observable size of the Earth and the Moon from your position (as you get farther from Earth it looks smaller; as you get closer to the Moon it looks bigger). Your observations aren't perfect, but you remember from CS 3600 that a Dynamic Bayesian Network can be used to estimate unobservable features—like your position in space—from imperfect observations. This is the Dynamic Bayesian Network you come up with, along with the conditional probability tables (not shown):



You decide to implement a particle filter to estimate your future position. This DBN is a bit different from the ones studied in class. What step in the particle filtering algorithm has to change to account for this network, and how must it be changed? You do not have to derive any equations.

In this case, having a history of evidence might help with identifying your position. Therefor I think this scenario is different since using a first-order Markov assumption may make it difficult to pinpoint your location. I would suggest that maybe a second order or higher Markov assumption be implemented to identify your location more accurately. So, in the particle filtering algorithm, when weighting the particles you would want to consider both sensor evidence history and position history. Position history is also important since it is likely that you are just following a trajectory and it is unlikely that you suddenly switch directions.

Question 7. (1 point) Someone develops an AI system that takes an image of you and makes it look more professional so that you can add it your LinkedIn account or résumé. For example, it will make your clothes look more expensive, remove blemishes and dirt from your face, make your hair look professionally and expensively groomed, and change the background to look like you are sitting in a private office. But this service needs a very big server and costs with high energy and maintenance costs. Thus the service costs $500 for a set of 5 pictures. What principle from our class discussion on societal implications must be considered, and why.

I think the societal implication that needs to be considered is fairness. This AI would only be only accessible to the people who could afford it an potentially give them an unfair advantage in professional opportunities. Additionally, it may give recruiters a false sense of who the person really is and can lead to potential misrepresentation of the applicant. This could lead to major misunderstandings and possible legal action further down the hiring process due to there not being complete transparency. Finally it seems like it is unfair to use a very big server that consumes large amounts of energy for something as trivial as a LinkedIn picture and isn't even readily open for everyone to use.