UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

CS440/ECE448 Artificial Intelligence

Conflict Exam 3

Spring 2023

May 7, 2023

Your Name: _____

Your NetID: _____

Instructions

- Please write your name on the top of every page.
- Have your ID ready; you will need to show it when you turn in your exam.
- This will be a CLOSED BOOK, CLOSED NOTES exam. You are permitted to bring and use only one 8.5x11 page of notes, front and back, handwritten or typed in a font size comparable to handwriting.
- No electronic devices (phones, tablets, calculators, computers etc.) are allowed.
- SHOW YOUR WORK. Correct answers derivation may not receive full credit if you don't show your work.
- Make sure that your answer includes only the variables that it should include, but DO NOT simplify explicit numerical expressions. For example, the answer $x = \frac{1}{1+\exp(-0.1)}$ is MUCH preferred (much easier for us to grade) than the answer $x = 0.524979$.

Possibly Useful Formulas

$$P(X=x|Y=y)P(Y=y) = P(Y=y|X=x)P(X=x)$$
$$P(X=x) = \sum_y P(X=x,Y=y)$$
$$E\left[f(X,Y)\right] = \sum_{x,y} f(x,y)P(X=x,Y=y)$$

$$\textbf{Precision,Recall} = \frac{TP}{TP+FP}, \frac{TP}{TP+FN}$$
$$\textbf{MPE=MAP:}\quad f(x) = \arg\max\left(\log P(Y=y) + \log P(X=x|Y=y)\right)$$
$$\textbf{Naive Bayes:}\quad P(X=x|Y=y) \approx \prod_{i=1}^{n} P(W=w_i|Y=y)$$

$$\textbf{Laplace Smoothing:}\quad P(X=x|Y=y) = \frac{\text{Count}(X=x,Y=y)+k}{\text{Count}(Y=y)+k|X|},\ \ |X| = \#\text{ possible distinct values of } X$$
$$\textbf{Fairness:}\quad P(Y|A) = \frac{P(Y|\hat{Y},A)P(\hat{Y}|A)}{P(\hat{Y}|Y,A)}$$

$$\textbf{Linear Regression:}\quad \varepsilon_i = f(x_i) - y_i = b + w@x_i - y_i$$
$$\textbf{Mean Squared Error:}\quad \text{MSE} = \frac{1}{n}\sum_{i=1}^{n} \varepsilon_i^2$$
$$\textbf{Linear Classifier:}\quad f(x) = \arg\max_k w_k@x + b$$

$$\textbf{Cross-Entropy:}\quad \mathscr{L} = -\frac{1}{n}\sum_{i=1}^{n} \log f_{y_i}(x_i)$$
$$\textbf{Softmax:}\quad \operatorname*{soft\,max}_c(w@x+b) = \frac{\exp(w_c@x+b_c)}{\sum_{k=0}^{V-1}\exp(w_k@x+b_k)}$$
$$\textbf{Softmax Error:}\quad \varepsilon_{i,c} = \begin{cases} f_c(x_i)-1 & c=y_i \\ f_c(x_i)-0 & \text{otherwise} \end{cases}$$
$$\textbf{Gradient Descent:}\quad w \leftarrow w - \eta\nabla_w\mathscr{L}$$

$$\textbf{Neural Net:}\quad h = \text{ReLU}\left(b_0 + w_0@x\right),\quad f = \operatorname*{soft\,max}\left(b_1 + w_1@h\right)$$
$$\textbf{Back-Propagation:}\quad \frac{\partial\mathscr{L}}{\partial h_j} = \sum_k \frac{\partial\mathscr{L}}{\partial f_k}\times\frac{\partial f_k}{\partial h_j},\quad \frac{\partial\mathscr{L}}{\partial w_{0,k,j}} = \frac{\partial\mathscr{L}}{\partial h_k}\times\frac{\partial h_k}{\partial w_{0,k,j}}$$

$$\textbf{Consistent Heuristic:}\quad h(p) \le d(p,r) + h(r)$$
$$\textbf{Alpha-Beta Max Node:}\quad v = \max(v,\text{child});\quad \alpha = \max(\alpha,\text{child})$$
$$\textbf{Alpha-Beta Min Node:}\quad v = \min(v,\text{child});\quad \beta = \min(\beta,\text{child})$$
$$\textbf{Variance Network:}\quad \mathscr{L} = \frac{1}{n-1}\sum_{i=1}^{n}\left(f_2(x_i) - \left(f_1(x_i)-x_i\right)^2\right)^2$$

$$\textbf{Unification:} \quad U = S(P) = S(Q);\ U \Rightarrow \exists x : Q;\ U \Rightarrow \exists x : P$$

$$\textbf{Bayes Rule:} \quad P(Y=y|X=x) = \frac{P(X=x|Y=y)P(Y=y)}{\sum_{y'} P(X=x|Y=y')P(Y=y')}$$

$$\textbf{Unnormalized Relevance:} \quad \tilde{R}(f_c, x_d) = \frac{\partial f_c}{\partial x_d} x_d f_c$$

$$\textbf{Normalized Relevance:} \quad R(f_c, x_d) = \frac{\frac{\partial f_c}{\partial x_d} x_d}{\sum_{d'} \frac{\partial f_c}{\partial x_{d'}} x_{d'}} f_c$$

$$\textbf{Softmax:} \quad \operatorname*{softmax}_j(e) = \frac{\exp(e_j)}{\sum_k \exp(e_k)}$$

$$\textbf{Softmax Deriv:} \quad \frac{\partial \operatorname{softmax}_m(e)}{\partial e_n} = \operatorname*{softmax}_m(e)\delta[m-n] - \operatorname*{softmax}_m(e)\operatorname*{softmax}_n(e), \quad \delta[m-n] = \begin{cases} 1 & m = n \\ 0 & m \neq n \end{cases}$$

$$\textbf{Viterbi:} \quad v_t(j) = \max_i v_{t-1}(i) a_{i,j} b_j(x_t)$$

$$\textbf{Transformer:} \quad c_i = \operatorname{softmax}(q_i @ k^T) @ v$$

$$\textbf{Pinhole Camera:} \quad \frac{x'}{f} = -\frac{x}{z}, \quad \frac{y'}{f} = -\frac{y}{z}$$

$$\textbf{Convolution:} \quad w_{k,l} * x_{k,l} = \sum_i \sum_j w_{k-i,l-j} x_{i,j}$$

$$\textbf{Kalman Prediction :} \quad \mu_{t|t-1} = \mu_{t-1|t-1} + \mu_\Delta, \quad \sigma^2_{t|t-1} = \sigma^2_{t-1|t-1} + \sigma^2_\Delta$$

$$\textbf{Kalman Gain :} \quad k_t = \frac{\sigma^2_{t|t-1}}{\sigma^2_{t|t-1} + \sigma^2_\varepsilon}, \quad \sigma^2_{t|t} = \sigma^2_{t|t-1}(1-k_t)$$

$$\textbf{Kalman Update:} \quad \mu_{t|t} = \mu_{t|t-1} + k_t\left(x_t - \left(\mu_{t|t-1} + \mu_\varepsilon\right)\right)$$

$$\textbf{Bellman Equation:} \quad U(s) = R(s) + \gamma \max_a \sum_{s'} P(s'|s,a) U(s')$$

$$\textbf{Value Iteration:} \quad U_i(s) = R(s) + \gamma \max_a \sum_{s'} P(s'|s,a) U_{i-1}(s')$$

$$\textbf{Policy Evaluation:} \quad U_i(s) = R(s) + \gamma \sum_{s'} P(s'|s,\pi_i(s)) U_i(s')$$

$$\textbf{Policy Improvement:} \quad \pi_{i+1}(s) = \arg\max_a R(s) + \gamma \sum_{s'} P(s'|s,a) U_i(s')$$

$$\textbf{Q-Learning:} \quad Q_{t+1}(s,a) = Q_t(s,a) + \alpha\left(Q_{\text{local}}(s,a) - Q_t(s,a)\right)$$

$$\textbf{TD Learning:} \quad Q_{\text{local}}(s_t, a_t) = R(s_t) + \gamma \max_{a'} Q_t(s_{t+1}, a')$$

$$\textbf{SARSA:} \quad Q_{\text{local}}(s_t, a_t) = R(s_t) + \gamma Q_t(s_{t+1}, a_{t+1})$$

$$\textbf{Imitation Learning:} \quad \mathscr{L} = -\log \pi_a(s)$$

$$\textbf{Deep Q Learning:} \quad \mathscr{L} = \frac{1}{2}\left(Q_t(s,a) - Q_{\text{local}}(s,a)\right)^2$$

$$\textbf{Actor-Critic:} \quad \mathscr{L} = -\sum_a \pi_a(s) Q(s,a)$$

$$\textbf{Inverse Kinematics:} \quad \mathscr{C}_{\text{obs}} = \{q : \exists b : \phi_b(q) \in \mathscr{W}_{\text{obs}}\}$$

Question 1 ***(7 points)***

Suppose that a classifier is being designed to predict whether a particular object is of class $Y =$a or class $Y =$b, based on observation that the object has features $X_1 = x_1$ and $X_2 = x_2$. The minimum probability of error (MPE) classifier has the following decision rule:

$$\text{Choose } Y = \text{a if and only if } P(Y = \text{a}|X_1 = x_1, X_2 = x_2) > P(Y = \text{b}|X_1 = x_1, X_2 = x_2) \tag{1}$$

Most practical implementations of MPE classifiers use the following decision rule, which is slightly different in form:

$$\text{Choose } Y = \text{a if and only if } P(Y = \text{a}, X_1 = x_1, X_2 = x_2) > P(Y = \text{b}, X_1 = x_1, X_2 = x_2) \tag{2}$$

(a) Prove that Rule 1 and Rule 2 choose $Y = a$ for exactly the same values of x_1 and x_2.

Solution: Using the definition of probability, we can rewrite Rule 1 as

$$\frac{P(Y = \text{a}, X_1 = x_1, X_2 = x_2)}{P(X_1 = x_1, X_2 = x_2)} > \frac{P(Y = \text{b}|X_1 = x_1, X_2 = x_2)}{P(X_1 = x_1, X_2 = x_2)}$$

Since the denominator is the same on both sides of the inequality, it can be deleted, resulting in Rule 2.

(b) Write the naïve Bayes approximation of Rule 2. Under what circumstances does the naïve Bayes approximation choose $Y = a$ for exactly the same set of x_1 and x_2 as Rule 2 (specify the name of the relationship between X_1, X_2, and Y that must hold)?

Solution: Choose $Y = a$ if and only if:

$$P(X_1 = x_1|Y = \text{a})P(X_2 = x_2|Y = \text{a})P(Y = \text{a}) > P(X_1 = x_1|Y = \text{b})P(X_2 = x_2|Y = \text{b})P(Y = \text{b})$$

This rule is the same as Rule 2 if X_1 and X_2 are conditionally independent given Y.

Question 2 ***(6 points)***

Imagine a two-layer neural net with input vector x, output vector f, and with the following architecture:

$$
\begin{aligned}
f_k &= \text{soft}\max(p) \\
p_k &= \sum_j w_{2,k,j} h_j \\
h_j &= \text{ReLU}(z_j) \\
z_j &= \sum_i w_{1,j,i} x_i
\end{aligned}
$$

What is $\frac{\partial h_j}{\partial x_i}$? Write your answer in terms of the variables x_m, z_m, h_m, p_m, f_m, $w_{1,m,n}$, and/or $w_{2,m,n}$ for any values of m and n that you find to be convenient. Show the steps in the chain rule that are necessary to derive your answer.

Solution:

$$
\begin{aligned}
\frac{\partial h_j}{\partial x_i} &= \frac{\partial h_j}{\partial z_j} \frac{\partial z_j}{\partial x_i} \\
&= u(z_j)\, w_{1,j,i}
\end{aligned}
$$

Question 3 ***(6 points)***

You are searching a maze in which positions are denoted by their (x,y) coordinate pairs, and in which the start node is $s = (4,4)$ and the goal node is $g = (0,0)$. Each step consists of one move in the horizontal or vertical direction; diagonal steps are not allowed. Your friend proposes implementing an A* search algorithm using the following heuristic:

$$h((x,y)) = |\sin(x)| + |\sin(y)|$$

Prove that $h((x,y))$ is an admissible heuristic for this problem.

Solution: An admissible heuristic is defined by

$$h(p) \leq d(p, \text{Goal})$$

Since diagonal steps are not allowed in this problem, and since the goal is $(0,0)$, we know that

$$|x| + |y| \leq d(p, \text{Goal})$$

But from calculus, we know that

$$h((x,y)) = |\sin(x)| + |\sin(y)| \leq |x| + |y| \leq d(p, \text{Goal})$$

Therefore $h((x,y))$ is admissible.

Question 4 ***(6 points)***

Consider the problem of trying to prove that Chicago is a city. Your goalset is the theorem:

$$\mathscr{G} = \{\text{CITY}(\text{chicago})\}$$

In the attempt to prove your goalset, you will use forward chaining, starting from the following fact database:

$$\mathscr{D} = \left\{ \begin{array}{ll} P_1: & \neg\text{COUNTRY}(x) \wedge \text{POPULATION}(x,y) \wedge \text{LARGER}(y,10000) \Rightarrow \text{CITY}(x) \\ P_2: & \text{POPULATION}(\text{chicago}, 7000000) \\ P_3: & \text{LARGER}(7000000, 10000) \\ P_4: & \text{PART-OF}(u,v) \wedge \text{COUNTRY}(v) \Rightarrow \neg\text{COUNTRY}(u) \\ P_5: & \text{PART-OF}(\text{chicago}, \text{usa}) \\ P_6: & \text{COUNTRY}(\text{usa}) \end{array} \right\}$$

Suppose that the first forward-chaining step determines that the consequent of P_4 can be unified with one of the antecedents of P_1. What new fact is added to the database as a result? What is the substitution dictionary that makes insertion of this new fact possible? Be sure that the variable names in your new fact are normalized so that they will not be confused with variables elsewhere in the database.

Solution: The new fact is

$$\text{PART-OF}(a,b) \wedge \text{COUNTRY}(b) \wedge \text{POPULATION}(a,c) \wedge \text{LARGER}(c,10000) \Rightarrow \text{CITY}(a)$$

The substitution dictionary is

$$\{x:a, y:c, u:a, v:b\}$$

Question 5 ***(6 points)***

Consider an HMM with two possible states ($Y \in \{1,2\}$), two possible observations ($X \in \{\text{blue}, \text{orange}\}$). Define the transition and observation probabilities to be $a_{i,j} = P(Y_{t+1} = j | Y_t = i)$ and $b_{j,k} = P(X_t = k | Y_t = j)$, where we use $X_t = 1$ and $X_t = 2$ to mean $X_t = \text{blue}$ and $X_t = \text{orange}$, respectively (alphabetical order). Suppose that these probabilities are given by the following matrices:

$$\begin{bmatrix} a_{1,1} = 0.2 & a_{1,2} = 0.8 \\ a_{2,1} = 0.7 & a_{2,2} = 0.7 \end{bmatrix}, \quad \begin{bmatrix} b_{1,1} = 0.1 & b_{1,2} = 0.9 \\ b_{2,1} = 0.6 & b_{2,2} = 0.4 \end{bmatrix}$$

What is $P(Y_t = 2 | Y_{t-1} = 1, X_t = 2)$? Be sure to express your answer in terms of the variables $a_{m,n}$ and $b_{m,k}$, for appropriate values of m and k, before you substitute in the provided numbers.

Solution:

$$\begin{aligned} P(Y_t = 2 | Y_{t-1} = 1, X_t = 2) &= \frac{P(Y_t = 2, X_t = 2 | Y_{t-1} = 1)}{P(X_t = 2 | Y_{t-1} = 1)} \\ &= \frac{a_{1,2} b_{2,2}}{a_{1,2} b_{2,2} + a_{1,1} b_{1,2}} \\ &= \frac{(0.8)(0.4)}{(0.8)(0.4) + (0.2)(0.9)} \end{aligned}$$

Question 6 ***(7 points)***

In a photograph of a hillside, you notice a road going straight up the hill. In the photograph, the two sides of the road seem to converge toward the vanishing point $(x', y') = (0, a)$. Write an equation for the height of the road, y, as a function of its distance from the camera, z. Your equation should be in terms of the variables a, y, z, the focal length f, and one more parameter that has not been mentioned in this problem statement. What is the parameter you must invent that has not been mentioned in this problem statement, and why?

Solution: The pinhole camera equations are

$$\frac{x'}{f} = -\frac{x}{z}, \quad \frac{y'}{f} = -\frac{y}{z}$$

In the limit as $z \to \infty$, we have that

$$\frac{y'}{f} = -\lim_{z\to\infty} \frac{y}{z} = a$$

if and only if

$$y = -\left(\frac{a}{f}\right) z + y_0$$

where y_0 is a constant offset, distinguishing it from all other parallel lines that converge to the point (x', y'). The constant offset must be invented because the information in the problem statement is insufficient to determine it, because $\lim_{z\to\infty} \frac{y_0}{z} = 0$ for any constant y_0.

Question 7 ***(7 points)***

Imagine a two-layer CNN that computes its second-layer output, $y_{m,n}$, from its first-layer hidden nodes $h_{k,l}$, which are computed in turn from the input pixels $x_{i,j}$ according to:

$$y_{m,n} = \sum_{k,l} w_{2,k,l} h_{m-k,n-l}$$
$$h_{k,l} = \text{ReLU}(z_{k,l})$$
$$z_{k,l} = \sum_{i,j} w_{1,i,j} x_{k-i,l-j}$$

What is $\frac{\partial y_{m,n}}{\partial x_{i,j}}$? Write your answer in terms of $y_{a,b}, h_{a,b}, z_{a,b}, x_{a,b}, w_{2,a,b}$, and/or $w_{1,a,b}$ for any values of a, b that you find to be useful. Be sure to show the derivation of your answer using the chain rule.

Solution:

$$\frac{\partial y_{m,n}}{\partial z_{k,l}} = \sum_{k,l} \frac{\partial y_{m,n}}{\partial h_{k,l}} \frac{\partial h_{k,l}}{\partial z_{k,l}} \frac{\partial z_{k,l}}{\partial x_{i,j}}$$
$$= \sum_{k,l} w_{2,m-k,n-l} u(z_{k,l}) w_{1,k-i,l-j}$$

Question 8 ***(7 points)***

A robot is delivering drinks to people on the quad. The terrain is uncertain; it believes its current position and velocity to be $(x_{t|t}, y_{t|t})$ (meters) and (u, v) (meters/minute), respectively, but these estimates are uncertain, with variances of $(\sigma^2_{x,t|t}, \sigma^2_{y,t|t})$ and (σ^2_u, σ^2_v), respectively. Based on these estimates, where does the robot predict it will be one minute later, and what is the variance of that prediction? Assume that the robot's current estimated position is independent of its velocity, and that the x and y components of each are independent.

Solution: Since position is in meters, speed is in meters/minute, and the change in time is one minute, we can just add the position and the speed to give:

$$(x_{t+1|t}, y_{t+1|t}) = \left(x_{t|t} + u, y_{t|t} + v\right)$$

with variances of:

$$(\sigma^2_{x,t+1|t}, \sigma^2_{y,t+1|t}) = \left(\sigma^2_{x,t|t} + \sigma^2_u, \sigma^2_{y,t|t} + \sigma^2_v\right)$$

Question 9 ***(7 points)***

Consider an MDP with two states ($s \in \{0,1\}$), two actions ($a \in \{0,1\}$), with rewards of $R(0) = -10$ and $R(1) = 10$, and with the following transition probabilities:

s'	$P(s'\|s=0,a=0)$	$P(s'\|s=0,a=1)$	$P(s'\|s=1,a=0)$	$P(s'\|s=1,a=1)$
0	0.4	0.7	0.9	0.2
1	0.6	0.3	0.1	0.8

Consider trying to solve for the utilities of the two states, $U(0)$ and $U(1)$, assuming $\gamma = \frac{1}{2}$, using value iteration. Consider starting with the values $U_1(0) = R(0) = -10$, $U_1(1) = R(1) = 10$. Write two equations that could be solved to find the second-iteration values $U_2(0)$ and $U_2(1)$. Write your equations in terms of the variables $R(0)$, $R(1)$, $U_1(0)$, $U_1(1)$, $P(0|0,0)$, $P(1|0,0)$, $P(0|0,1)$, $P(1|0,1)$, $P(0|1,0)$, $P(1|1,0)$, $P(0|1,1)$, $P(1|1,1)$ and/or γ first, then substitute in the provided numerical values.

Solution:

$$U_2(0) = R(0) + \gamma \max_a \sum_{s'} P(s'|s=0,a) U_1(s')$$
$$U_2(1) = R(1) + \gamma \max_a \sum_{s'} P(s'|s=1,a) U_1(s')$$

With the values filled in, this is:

$$U_2(0) = -10 + \frac{1}{2}\max\left((0.4(-10)+0.6(10)), (0.7(-10)+0.3(10))\right) U_2(1) \quad = 10 + \frac{1}{2}\max\left((0.9(-10)+0.1(10)), (\right.$$

Question 10 ***(7 points)***

Consider an MDP with four states, $s \in \{0,1,2,3\}$. The rewards for these states are $R(0) = 0, R(1) = 1, R(2) = -10000, R(3) = 10000$. From any of these states, there are three possible actions, $a \in \{0,1,2\}$.

Consider a model-based reinforcement learning (RL) algorithm that is trying to learn the transition probabilities for this MDP. Unknown to the RL, the true transition probabilities are the same for every state, and they are:

	$P(s'\|s,a=0)$				$P(s'\|s,a=1)$				$P(s'\|s,a=2)$			
s'	0	1	2	3	0	1	2	3	0	1	2	3
$s=0$	0.5	0.5	0	0	0.99	0	0	0.01	0	0.99	0.01	0
$s=1$	0.5	0.5	0	0	0.99	0	0	0.01	0	0.99	0.01	0
$s=2$	0.5	0.5	0	0	0.99	0	0	0.01	0	0.99	0.01	0

$$P(s'|s,a=0) = \begin{cases} 0.5 & s'=0, \forall s \\ 0.5 & s'=1, \forall s \end{cases}$$

$$P(s'|s,a=1) = \begin{cases} 0.99 & s'=0, \forall s \\ 0.01 & s'=3, \forall s \end{cases}$$

$$P(s'|s,a=2) = \begin{cases} 0.99 & s'=1, \forall s \\ 0.01 & s'=2, \forall s \end{cases}$$

Suppose that a model-based RL algorithm uses an epsilon-first exploration vs. exploitation policy to try to learn these transition probabilities, using $N = \frac{1}{\varepsilon} = 5$ trials per state/action pair. Consider the probable outcome of this experiment. When the exploration is over and exploitation begins, which action is most likely to be chosen as the optimal action? Will the RL continue to view this action as optimal, even after thousands of trials? Is there any action that will probably never be chosen after the initial exploration phase?

Solution:

- When exploitation begins, action $a = 1$ will probably be considered optimal. Action $a = 0$ leads to state 1 only about half the time, but during the 5 initial trials, action $a = 1$ will probably lead to state 1 every time, therefore action $a = 1$ will seem to have the bigger payoff.
- After thousands of trials, the RL will have enough observations of action $a = 1$ to see that it leads to a huge loss ($-10,000$) about once every hundred trials. Therefore, the RL will switch over to considering action $a = 0$ the optimal action.
- The RL will probably never use action $a = 2$ after the initial exploration phase. The initial exploration phase will probably never discover that action $a = 2$ sometimes leads to the high-payoff state. If this is not discovered during the initial exploration phase, then the algorithm will never choose action $a = 2$ during the exploitation phase, so it will never learn this fact about action $a = 2$.

Question 11 ***(7 points)***

Consider an MDP with two actions ($a \in \{R, L\}$), and three states ($s \in \{0, 1, 2\}$). The agent has been accumulating its experiences in an experience replay buffer. The buffer now contains (state,action,reward,new state) (s, a, r, s') tuples for six randomly selected trials, as shown in this table:

Trial ID	s	a	r	s'
1	1	R	10	2
2	1	R	10	0
3	0	L	20	1
4	0	L	20	1
5	0	L	20	0
6	2	R	5	1

Using Laplace smoothing with a smoothing coefficient of k, what are $P(s'|s=1, a=R)$ for $s' \in \{0, 1, 2\}$? Write down the general formula for Laplace smoothing, then fill in the numerical values for the problem.

Solution: The general formula for Laplace smoothing is

$$P(X = x|Y = y) = \frac{\text{Count}(X = x, Y = y) + k}{\text{Count}(Y = y) + k|X|},$$

where $|Y|$ is the number of logically distinct possible values of X. For this problem, the results are:

$$P(s' = 0|s = 1, a = R) = \frac{1+k}{2+3k}$$
$$P(s' = 1|s = 1, a = R) = \frac{1+k}{2+3k}$$
$$P(s' = 2|s = 1, a = R) = \frac{k}{2+3k}$$

Question 12 ***(7 points)***

A cat lives in a two-room apartment, and its current state is well-summarized by specifying the room it currently occupies, $s \in \{1,2\}$. In each room, the cat can choose one of two possible actions, $a \in \{\text{pounce}, \text{sleep}\}$. Its current estimates of the quality of each action are:

	$Q_t(s, a = \text{pounce})$	$Q_t(s, a = \text{sleep})$
$s = 1$	4	6
$s = 2$	8	3

At time t, the cat is in room $s_t = 1$, and receives a reward of $R(1) = 2$ kitty treats. The cat decides to pounce ($a_t = \text{pounce}$), and finds itself in room 2 at time $t+1$. Having found itself in room $s_{t+1} = 2$, the cat decides to sleep ($a_{t+1} = \text{sleep}$). Find two different ways in which the cat might update $Q_t(s,a)$ to compute $Q_{t+1}(s = 1, a = \text{pounce})$. Specifically, find $Q_{t+1}(s = 1, a = \text{pounce})$ using both TD-learning and SARSA. Both updates should be written as functions of the learning rate, α, and the discount factor, γ; there should be no other variables on the right-hand-side of your answer.

Solution: Using TD-learning and SARSA, respectively, Qlocal is:

$$\textbf{TD Learning:} \quad Q_{\text{local}}(s_t, a_t) = R(s_t) + \gamma \max_{a'} Q_t(s_{t+1}, a')$$

$$\textbf{SARSA:} \quad Q_{\text{local}}(s_t, a_t) = R(s_t) + \gamma Q_t(s_{t+1}, a_{t+1})$$

Plugging in the numbers from this problem, we have

$$\textbf{TD Learning:} \quad Q_{\text{local}}(1, \text{pounce}) = 2 + 8\gamma$$

$$\textbf{SARSA:} \quad Q_{\text{local}}(1, \text{pounce}) = 2 + 3\gamma$$

The resulting Q-learning updates are

$$\textbf{TD Learning:} \quad Q_{t+1}(7,2) = (1-\alpha)4 + \alpha(2 + 8\gamma)$$

$$\textbf{SARSA:} \quad Q_{\text{local}}(7,2) = (1-\alpha)4 + \alpha(2 + 3\gamma)$$

Question 13 ***(7 points)***

The cat decides to implement deep-Q learning so that it can take advantage of much more information: its new state variable is a vector of six measurements including the (x,y) position of its human servant, the amount of food in its bowl, the temperature of the warm spot in front of the window, and its own (x,y) position in the apartment. Given this state vector, the cat's neural network computes a vector of hidden nodes $h(s) = [h_0(s), \ldots, h_{n-1}(s)]$, then computes two outputs corresponding to the two possible actions, $a \in \{\text{pounce}, \text{sleep}\}$. The two outputs of the neural network are

$$Q_t(s,a) = \sum_{i=0}^{n-1} w_{t,a,i} h_i(s), \quad a \in \{\text{pounce}, \text{sleep}\},$$

where $w_{t,a,i}$ are weights that are trained using stochastic gradient descent:

$$w_{t+1,a,i} = w_{t,a,i} - \alpha \frac{\partial \mathscr{L}_t}{\partial w_{t,a,i}} \tag{3}$$

Suppose the cat measures the state vector s_t, receives a reward of $r_t = 2$, decides on the action $a_t =$ pounce, and then measures the resulting state vector s_{t+1} at time $t+1$. On the basis of these measurements, propose a loss function $\mathscr{L}_t$ whose derivative could be used to update the neural network as shown in Eq. 3. Specify exactly the way in which your loss function depends on the discount factor γ and on the neural network outputs that might be computed from input vectors s_t ($Q_t(s_t,0)$ and/or $Q_t(s_t,1)$) and/or from input vector s_{t+1} ($Q_t(s_{t+1},0)$ and/or $Q_t(s_{t+1},1)$).

Solution: The action a_{t+1} is not specified, so we can't use SARSA, so let's use TD-learning:

$$Q_{\text{local}}(s_t, a_t = \text{pounce}) = R(s_t) + \gamma \max_{a'} Q_t(s_{t+1}, a') \qquad = 2 + \gamma \max_{a'} Q_t(s_{t+1}, a')$$

A useful loss function might be the squared distance between $Q_t(s_t, \text{pounce})$ and $Q_{\text{local}}(s_t, \text{pounce})$, which is

$$\mathscr{L} = \frac{1}{2}\left(Q_t(s_t, \text{pounce}) - 2 - \gamma \max_{a'} Q_t(s_{t+1}, a')\right)^2$$

Question 14 ***(7 points)***

Recall that in actor-critic reinforcement learning, an agent in state s should choose action a with probability $\pi_a(s) = \text{softmax}_a(w@h(s))$, where s is the state vector, $h(s)$ is a vector of hidden nodes, and w is a weight matrix. The weight matrix w is trained to maximize the expected utility of state s, $E[Q(s,a)] = \sum_a \pi_a(s)Q(s,a)$, where $Q(s,a)$ is the quality of action a in state s as estimated by the critic network, and $\pi_a(s)$ is the probability of choosing action a. Now consider an actor-critic network trained using an epsilon-greedy policy. With probability ε, the actor will choose one of N actions uniformly at random. With probability $1-\varepsilon$, the actor will choose actions according to the probabilities given by the neural network, $\pi_a(s)$. Since the epsilon-greedy policy changes the probability of choosing action a, it also changes $E[Q(s,a)]$. What is the value of $E[Q(s,a)]$ that would apply to an actor-critic network using an epsilon-greedy policy?

Solution: The probability of choosing action a is now

$$P(a) = \frac{\varepsilon}{N} + (1-\varepsilon)\pi_a(s)$$

So

$$E[Q(s,a)] = \sum_a \left(\frac{\varepsilon}{N} + (1-\varepsilon)\pi_a(s)\right) Q(s,a)$$

Question 15 ***(7 points)***

A robot fire truck works in a workspace with a horizontal dimension (x) and a vertical dimension (y). The robot fire truck's base must remain at $y = 0$, but it can move its base back and forth to position $x = p$. The ladder can be raised to elevation angle θ ($0 \leq \theta \leq \pi$), and the ladder can be extended to length l. A point b meters along the ladder ($0 \leq b \leq l$) is thus located by the forward-kinematic function as

$$\phi_b\left(\begin{bmatrix} p \\ \theta \\ l \end{bmatrix}\right) = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} p + b\cos\theta \\ b\sin\theta \end{bmatrix}$$

In the workspace $\mathscr{W}$, there is an obstacle: a wall at $x = c$, which makes it impossible for any part of the robot to exist at any point $x \geq c$, where you may assume that $c > 0$. This can be written as:

$$\mathscr{W}_{\text{obs}} = \{(x,y) : x \geq c\}$$

Define $\mathscr{C}_{\text{obs}}$ to be the set of robot configurations $[p,\theta,l]$ that place any part of the robot's ladder within $\mathscr{W}_{\text{obs}}$. This can be written as

$$\mathscr{C}_{\text{obs}} = \{(p,\theta,l) : P\},$$

where P is some inequality or disjunction of inequalities in terms of p, θ and l. Find P.

Solution: The robot hits the obstacle if any part of its ladder is in $\mathscr{W}_{\text{obs}}$. Literally, we can write this as:

$$P = (\exists b \in [0,l] : p + b\cos\theta \geq c)$$

It is also acceptable to note that we only hit the obstacle if either the lower or upper end of the ladder are inside the obstacle. These two inequalities are a little bit simpler, they are just:

$$P = (p \geq c) \vee (p + l\cos\theta \geq c)$$

THIS PAGE IS SCRATCH PAPER