- LIFO (last-in, first-out) = Depth-First Search (DFS):
  - the next node you expand will always be the one **<u>mos</u>** **<u>recently</u>** added to the frontier.
- FIFO (first-in, first-out) = Breadth-First Search (BFS):
  - the next node you expand will always be the one **<u>leas</u>** **<u>recently</u>** added to the frontier.
- PriorityQueue (lowest-cost, first-out) = Uniform Cos Search (UCS, a.k.a. Dijkstra's algorithm):
  - the next node you expand will always be the one witl **<u>lowest cost</u>**

trategies are evaluated along the following riteria:

Completeness: does it always find a solution one exists? • Optimality: does it always find least-cost solution?

Time complexity: number of nodes enerated

Space complexity: maximum number of odes in memory

Time and space complexity are measured in erms of

b: maximum branching factor of the search ree

d: depth of the optimal solution

m: maximum length of any path in the state pace (may be infinite)

Question 29 *(0 points)*

A particular hidden Markov model (HMM) has state variable X_t, and observation variables E_t, where t denotes time. Suppose that this HMM has two states, $X_t \in \{0,1\}$, and three possible observations, $E_t \in \{0,1,2\}$. The initial state probability is $P(X_1=1)=0.3$. The transition and observation probability matrices are

X_{t-1}	$P(X_t=1\|X_{t-1})$	X_t	$P(E_t=0\|X_t)$	$P(E_t=1\|X_t)$
0	0.6	0	0.4	0.1
1	0.4	1	0.1	0.6

Suppose that, in a particular test of the HMM, the observation sequence is

$$\{E_1,E_2\}=\{2,1\}$$

(a) What is the total probability $P(E_1=2,E_2=1)$?

Solution:

$$P(E_1=2,E_2=1)=\sum_{X_1,X_2} P(X_1)P(E_1=2|X_1)P(X_2|X_1)P(E_2=1|X_2)$$
$$=(0.7)(0.5)(0.4)(0.1)+(0.7)(0.5)(0.6)(0.6)+(0.3)(0.3)(0.6)(0.1)+(0.3)(0.3)(0.4)(0.6$$

(b) If it is observed that $X_2=1$, what is the most likely value of X_1? In other words, what is $\arg\max_{X_1} P(X_1,E_1=2,X_2=1,E_2=1)$?

Solution:

$$\arg\max_{X_1} P(X_1,E_1=2,X_2=1,E_2=1)=\arg\max_{X_1} P(X_1)P(E_1=2|X_1)P(X_2=1|X_1)$$
$$=\arg\max_{0,1}((0.7)(0.5)(0.6),(0.3)(0.3)(0.4))$$
$$=0$$

(a) What is the joint probability $P(E_1=2,X_2=1,E_2=1)$?

Solution:

$$P(E_1=2,X_2=1,E_2=1)=\sum_{X_1} P(X_1)P(E_1=2|X_1)P(X_2=1|X_1=0)P(E_2=1|X_2=1)$$
$$=(0.7)(0.5)(0.6)(0.6)+(0.3)(0.3)(0.4)(0.6)$$

(b) If it is observed that $X_2=0$, what is the most likely value of X_1? In other words, what i: $\arg\max_{X_1} P(X_1,E_1=2,X_2=0,E_2=1)$?

Solution:

$$\arg\max_{X_1} P(X_1,E_1=2,X_2=0,E_2=1)=\arg\max_{X_1} P(X_1)P(E_1=2|X_1)P(X_2=0|X_1)$$
$$=\arg\max((0.7)(0.5)(0.4),(0.3)(0.3)(0.6))$$
$$=0$$

(a) What is the joint probability $P(X_1=1,E_1=2,X_2=0)$?

Solution:

$$P(X_1=1,E_1=2,X_2=0)=P(X_1=1)P(E_1=2|X_1=1)P(X_2=0|X_1=1)$$
$$=(0.3)(0.3)(0.6)$$

(b) What is the probability of the most likely state sequence ending in $X_2=0$? In other words, what is $\max_{X_1} P(X_1,E_1=2,X_2=0,E_2=1)$?

Solution:

$$\max_{X_1} P(X_1,E_1=2,X_2=0,E_2=1)=\max_{X_1} P(X_1)P(E_1=2|X_1)P(X_2=0|X_1)P(E_2=1|X_2=0)$$
$$=\max((0.7)(0.5)(0.4)(0.1),(0.3)(0.3)(0.6)(0.1))$$
$$=(0.7)(0.5)(0.4)(0.1)$$

(a) What is the joint probability $P(X_1=0,E_1=2,E_2=1)$?

Solution:

$$P(X_1=0,E_1=2,E_2=1)=\sum_{X_2} P(X_1=0)P(E_1=2|X_1=0)P(X_2|X_1=0)P(E_2=1|X_2)$$
$$=(0.7)(0.5)(0.4)(0.1)+(0.7)(0.5)(0.6)(0.6)$$

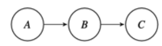
(b) What is the probability of the most likely state sequence ending in $X_2=1$? In other words, what is $\max_{X_1} P(X_1,E_1=2,X_2=1,E_2=1)$?

Solution:

$$\max_{X_1} P(X_1,E_1=2,X_2=1,E_2=1)=\max_{X_1} P(X_1)P(E_1=2|X_1)P(X_2=1|X_1)P(E_2=1|X_2=1)$$
$$=\max((0.7)(0.5)(0.6)(0.6),(0.3)(0.3)(0.4)(0.6))$$
$$=(0.7)(0.5)(0.6)(0.6)$$

Question 16 *(0 points)*

Consider the following Bayes network (all variables are binary):



$P(A)=0.8$

A	P(B|A)
False	0.7
True	0.3

B	P(C|B)
False	0.5
True	0.7

(a) What is $P(C)$? Write your answer in numerical form, but you don't need to simplify.

Solution:

$$P(C)=P(\neg A,\neg B,C)+P(\neg A,B,C)+P(A,\neg B,C)+P(A,B,C)$$
$$=(0.2)(0.3)(0.5)+(0.2)(0.7)(0.7)+(0.8)(0.7)(0.5)+(0.8)(0.3)(0.7)$$

(b) What is $P(A|B=True,C=True)$? Write your answer in numerical form, but you don't need to simplify.

Solution:

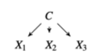
$$P(A|B,C)=\frac{P(A,B,C)}{P(A,B,C)+P(\neg A,B,C)}$$
$$=\frac{(0.8)(0.3)(0.7)}{(0.8)(0.3)(0.7)+(0.2)(0.7)(0.7)}$$

Question 20 *(0 points)*

We have a bag of three biased coins, a, b, and c, with probabilities of coming up heads of 20%, 60%, and 80%, respectively. One coin is drawn randomly from the bag (with equal likelihood of drawing each of the three coins), and then the coin is flipped three times to generate the outcomes X1, X2, and X3.

(a) Draw the Bayesian network corresponding to this setup and define the necessary conditional probability tables (CPTs).

Solution: You need an intermediate variable, $C\in\{a,b,c\}$, to specify which coin is drawn, then the graph is



and the CPTs are

C	$P(C)$	$P(X_1=H\|C)$	$P(X_2=H\|C)$	$P(X_3=H\|C)$
a	1/3	0.2	0.2	0.2
b	1/3	0.6	0.6	0.6
c	1/3	0.8	0.8	0.8

(b) Calculate which coin was most likely to have been drawn from the bag if the observed flips come out heads twice and tails once.

Solution:

$$P(C=a,HHT)=(0.2)(0.2)(0.8)/3=32/3000$$
$$P(C=b,HHT)=(0.6)(0.6)(0.4)/3=144/3000$$
$$P(C=c,HHT)=(0.8)(0.8)(0.2)/3=128/3000$$

The maximum-posterior-probability event is also the maximum-joint-probability event, which is the event $C=b$.

Question 5 *(7 points)*

Consider a neural network with three input nodes, $x=[x_0,x_1,x_2]$, three hidden nodes, $h=[h_0,h_1,h_2]$, and three output nodes, $f=[f_0,f_1,f_2]$, related by

$$h=\mathrm{ReLU}(w_0@x)$$
$$f=\mathrm{softmax}(w_1@h)$$

where

$$w_0=\begin{bmatrix}0.8 & 0.3 & 0.5\\ -0.4 & -0.9 & -0.5\\ -0.2 & -0.8 & 0.7\end{bmatrix},\quad w_1=\begin{bmatrix}0.4 & -0.6 & 0.6\\ -0.7 & 0.2 & -0.3\\ 0.9 & -0.1 & 0.1\end{bmatrix}$$

Suppose $x=[1,0,-1]$. It can be computed that, in this case, $f=[0.33,0.26,0.41]$. What is the unnormalized relevance of the hidden node h_0 to the output f_2?

Solution: Unnormalized relevance is

$$\tilde{R}(f_2,h_0)=\frac{\partial f_2}{\partial h_0}h_0f_2$$

We already know that $f_2=0.41$. We need to find h_0 and $\frac{\partial f_2}{\partial h_0}$.

$$h=\begin{bmatrix}h_0\\ h_1\\ h_2\end{bmatrix}=\mathrm{ReLU}\left(\begin{bmatrix}0.8 & 0.3 & 0.5\\ -0.4 & -0.9 & -0.5\\ -0.2 & -0.8 & 0.7\end{bmatrix}\begin{bmatrix}1\\ 0\\ -1\end{bmatrix}\right)$$

Keeping just the first row of that equation, we have

$$h_0=\max(0,0.8-0.5)=0.3$$

Now for the derivative. We have that

$$\frac{\partial f_2}{\partial h_0}=\sum_j\frac{\partial\,\mathrm{softmax}_2(e)}{\partial e_j}\frac{\partial e_j}{\partial h_0}$$
$$=\sum_j\left(\mathrm{softmax}_2(e)\delta[2-j]-\mathrm{softmax}_2(e)\,\mathrm{softmax}_j(e)\right)w_{1,j,0}$$
$$=\sum_j(f_2\delta[2-j]-f_2f_j)w_{1,j,0}$$

Since the exam instructions tell us our answer can't include summations, we need to simplify that as

$$\frac{\partial f_2}{\partial h_0}=(f_2-f_2^2)w_{1,2,0}-f_2f_1w_{1,1,0}-f_2f_0w_{1,0,0}$$
$$=(0.41-(0.41)^2)(0.9)-(0.41)(0.26)(-0.7)-(0.41)(0.33)(04)$$

Putting it all together,

$$\tilde{R}(f_2,h_0)=((0.41-(0.41)^2)(0.9)-(0.41)(0.26)(-0.7)-(0.41)(0.33)(04))(0.3)(0.41)$$

Question 4 *(7 points)*

Suppose your burglar alarm has four modes: quiet ($A=0$), warning ($A=1$), alarm ($A=2$), and klaxon ($A=3$). Suppose the probability of an earthquake is $P(E=T)=0.01$, and the probability of a burglary is $P(B=T)=0.001$. Your neighbor, Jack, sends you a text message (event $J=T$) with a probability that depends only on the status of your alarm. Conditional probabilities for the variables A and J are given as the variables a through u in the following table:

	$a=0$	$a=1$	$a=2$	$a=3$
$P(A=a\|B=F,E=F)$	b	c	d	e
$P(A=a\|B=F,E=T)$	f	g	h	i
$P(A=a\|B=T,E=F)$	j	k	l	m
$P(A=a\|B=T,E=T)$	n	o	p	q
$P(J=T\|A=a)$	r	s	t	u

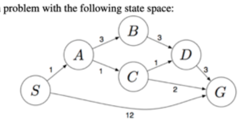
In terms of the parameters a through u, what is $P(B=T|A=3)$?

Solution:

$$P(B=T|A=3)=\frac{P(B=T,A=3)}{P(A=3)}$$
$$=\frac{\sum_{e'=F}^{T}P(B=T,E=e',A=3)}{\sum_{b'=F}^{T}P(B=b',E=e',A=3)}$$
$$=\frac{(0.001)(0.01q+0.99m)}{(0.001)(0.01q+0.99m)+(0.999)(0.01i+0.99e)}$$

Question 8 *(0 points)*

Consider the search problem with the following state space:



S denotes the start state, G denotes the goal state, and step costs are written next to each arc. Assume that ties are broken alphabetically (i.e., if there are two states with equal priority on the frontier, the state that comes first alphabetically should be visited first).

(a) What path would BFS return for this problem?

Solution: SG

(b) What path would DFS return for this problem?

Solution: SABDG

(c) What path would UCS return for this problem?

Solution: SACG

Question 1 *(0 points)*

Discuss the relative strengths and weaknesses of breadth-first search vs. depth-first search for AI problems.

Solution:

	BFS	DFS
Strength	Solution is guaranteed to be optimal. BFS is complete.	Can find goal faster than BFS if there are multiple solutions. Linear memory requirement ($O(bm)$).
Weakness	Exponential ($O(b^d)$) space complexity.	Solution not guaranteed to be optimal. Not complete. Computation is exponential in longest path ($O(b^m)$), rather than shortest path ($O(b^d)$).

Question 2 *(0 points)*

In the tree search formulation, why do we restrict step costs to be non-negative?

Solution: If the cost around any loop is negative, then the lowest-cost path is to take that loop an infinite number of times.

Question 3 *(0 points)*

What is the distinction between a world state and a search tree node?

Solution: A world state contains enough information to know (1) whether or not you've reached the goal, (2) what actions can be performed, (3) what will be the result of each action. A search tree node contains a pointer to the world state, plus a pointer to the parent node.

Question 4 *(0 points)*

How do we avoid repeated states during tree search?

Solution: By keeping a set of "explored states." If expanding a search node results in a state that has already been explored, we don't add it to the frontier.

Question 7 *(7 points)*

Suppose the the input to a transformer is the sequence of scalar values $v_t=\cos\left(\frac{t}{1000}\right)$, where $0\le t\le 999$. You are trying to find the context, c_i, for a query q_i whose inner product with the keys is

$$q_i@k_t=\begin{cases}0 & t\in\{250,251,252\}\\ -\infty & \text{otherwise}\end{cases}$$

Find the numerical value of c_i.

Solution: The transformer computes

$$c_i=\mathrm{softmax}\,q_i@k^T@v.$$

The t^{th} element of the softmax output is

$$\mathrm{softmax}_t(q_i@k)=\frac{\exp(q_i@k_t)}{\sum_{t'}\exp(q_i@k_{t'})}$$
$$=\begin{cases}\frac{\exp(0)}{3\exp(0)+997\exp(-\infty)} & t\in\{250,251,252\}\\ \frac{\exp(-\infty)}{3\exp(0)+997\exp(-\infty)} & \text{otherwise}\end{cases}$$
$$=\begin{cases}\frac{1}{3} & t\in\{250,251,252\}\\ 0 & \text{otherwise}\end{cases}$$

The context vector is therefore

$$c_i=\frac{1}{3}\cos\left(\frac{250}{1000}\right)+\frac{1}{3}\cos\left(\frac{251}{1000}\right)+\frac{1}{3}\cos\left(\frac{252}{1000}\right)$$

Question 6 *(7 points)*

You are a martial arts master, practicing blindfolded. Your opponent starts out either in front of you, to your left, behind you, or to your right. With each step, they either stay where they are, move 90° clockwise, or move 90° counter-clockwise, with equal probability. Each step (even if they don't move) makes a sound, but because of the echos, you only hear the correct direction of the sound with probability p; with probability $1-p$, you hear the sound from one of the three incorrect directions (each with equal probability). Suppose you hear the sound to your left, and then behind you ($X_0=L,X_1=B$). What is the probability that these observations occurred, and that your opponent is now behind you? That is, if your opponent's actual position is Y_t and their apparent position is X_t, what is $P(Y_1=B,X_0=L,X_1=B)$?

Solution:

$$P(Y_1=B,X_0=L,X_1=B)=\sum_{y_0}P(Y_0=y_0,X_0=L,Y_1=B,X_1=B)$$
$$=\sum_{y_0}\pi_{y_0}b_{y_0}(L)a_{y_0,B}b_B(B)$$

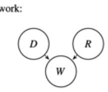
P(X₀=L, Y₀=y₀) · P(X₁=B, Y₁=B | Y₀=y₀)

$$=\left(\frac{1}{4}\right)p\left(\frac{1}{3}\right)p+\left(\frac{1}{4}\right)\left(\frac{1-p}{3}\right)\left(\frac{1}{3}\right)p+\left(\frac{1}{4}\right)\left(\frac{1-p}{3}\right)\left(\frac{1}{3}\right)p,$$

where the three possibilities correspond to $Y_0=L$, $Y_0=B$, and $Y_0=R$; from the position $Y_0=F$, it is not possible to move to $Y_1=B$.

Question 4 *(7 points)*

Consider the following Bayesian network:



Suppose that the model parameters are as follows:

$$P(D=d)=\frac{1}{2}\text{ for }d\in\{1,2\}$$
$$P(R=r)=\frac{1}{2}\text{ for }r\in\{1,2\}$$
$$P(W=T|D=d,R=r)=\begin{cases}\frac{2}{3} & d\ge r\\ \frac{1}{3} & d<r\end{cases}$$

What is $P(D=2|W=T)$?

Solution:

$$P(D=2|W=T)=\frac{\sum_{r=1}^{2}P(D=2,R=r,W=T)}{\sum_{d=1}^{2}\sum_{r=1}^{2}P(D=d,R=r,W=T)}$$
$$=\frac{\left(\frac{1}{2}\right)\left(\frac{1}{2}\right)\left(\frac{2}{3}\right)+\left(\frac{1}{2}\right)\left(\frac{1}{2}\right)\left(\frac{2}{3}\right)}{\left(\frac{1}{2}\right)\left(\frac{1}{2}\right)\left(\frac{2}{3}\right)+\left(\frac{1}{2}\right)\left(\frac{1}{2}\right)\left(\frac{2}{3}\right)+\left(\frac{1}{2}\right)\left(\frac{1}{2}\right)\left(\frac{2}{3}\right)+\left(\frac{1}{2}\right)\left(\frac{1}{2}\right)\left(\frac{1}{3}\right)}$$

If you wish, you can simplify the last formula to 4/7, but it's not required.

Question 6 *(7 points)*

Your apartment is haunted by a ghost. Like most ghosts, your ghost tends to sleep for several days at a time. Let $Y_t = T$ if the ghost is awake on day t. You can't see the ghost, but if the ghost is awake, your cat tends to hide under the bed; let $X_t = T$ if your cat is hiding under the bed on day t. Suppose that these probabilities are given by the following distribution, where a, b, c and d are arbitrary constants:

Y_{t-1}	$P(Y_t = T \mid Y_{t-1})$
F	a
T	b

Y_t	$P(X_t = T \mid Y_t)$
F	c
T	d

Suppose you know that the ghost was asleep on day 0 ($Y_0 = F$). You don't know whether or not it was awake on day 1, but you know that your cat hid under the bed ($X_1 = T$). In terms of a, b, c and/or d, find $P(Y_1 = T \mid Y_0 = F, X_1 = T)$.

Solution:

$$P(Y_1 = T \mid Y_0 = F, X_1 = T) = \frac{P(Y_1 = T, X_1 = T \mid Y_0 = F)}{P(X_1 = T \mid Y_0 = F)}$$
$$= \frac{ad}{ad + (1-a)c}$$

Question 7 *(7 points)*

Consider an HMM with state variables $Y_1, \ldots, Y_T$ and observations $X_1, \ldots, X_T$. Suppose that the model has the following parameters, where a, b, c and d are some arbitrary constants:

Y_{t-1}	$P(Y_t = 2 \mid Y_{t-1})$
1	a
2	b

Y_t	$P(X_t = 2 \mid Y_t)$
1	c
2	d

For a particular observation sequence $X_1 = x_1, \ldots, X_T = x_T$, define the Viterbi vertex probability to be

$$v_{j,t} = \max_{y_1,\ldots,y_{t-1}} P(Y_1 = y_1, X_1 = x_1, \ldots, Y_{t-1} = y_{t-1}, X_{t-1} = x_{t-1}, Y_t = j, X_t = x_t)$$

Suppose that $v_{j,t}$ has been calculated, and has been found to have the following values, where e and f are some arbitrary constants:

$$v_{1,t} = e$$
$$v_{2,t} = f$$

Furthermore, suppose that $x_{t+1} = 2$. In terms of the constants a, b, c, d, e, and/or f, what are $v_{1,t+1}$ and $v_{2,t+1}$?

Solution:

$$v_{1,t+1} = \max\left(e(1-a)c, f(1-b)c\right)$$
$$v_{2,t+1} = \max\left(ead, fbd\right)$$

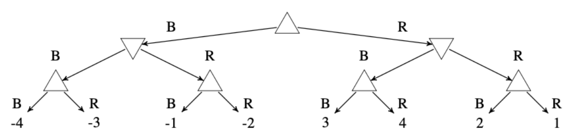
Question 13 *(0 points)*

What are the main challenges of adversarial search as contrasted with single-agent search? What are some algorithmic similarities and differences?

Solution: The biggest difference is that we are unaware of how the opponent(s) will act. Because of this our search cannot simply consider my own moves, it must also figure out how my opponent will act at each level, thus effectively doubling the number of levels over which I have to search. Since the number of levels is the exponent in the computational complexity, this makes computational complexity much harder.

Question 7 *(8 points)*

The following **minimax tree** shows all possible outcomes of the RED-BLUE game. In this game, Max plays first, then Min, then Max. Each player, when it's their turn, chooses either a blue stone (B) or a red stone (R); after three turns, Max wins the number of points shown (negative scores indicate a win for Min).



(a) (3 points) Max could be a Reflex Agent, following a set of predefined IF-THEN rules, and could still play optimally against Min, even if Min is not rational. To do so, Max needs just three rules of the form "If the stones already chosen are ____, then choose a ____ stone." Write those three rules in that form.

Solution:

- If no stones have been chosen, then choose a Red stone.
- If the stones already chosen are (R,B), then choose a Red stone.
- If the stones already chosen are (R,R), then choose a Blue stone.

(b) (2 points) Recall that an $\alpha - \beta$ search prunes the largest possible number of moves if there is extra information available to the players that permits them to evaluate the moves in the best possible order. IN GENERAL (not just for this game tree),

- In what order should the moves available to MAX be evaluated, in order to prune as many moves as possible?
- In what order should the moves available to MIN be evaluated, in order to prune as many moves as possible?

Solution:

- Moves available to MAX should be evaluated in order of descending value, starting with the highest-value move.
- Moves available to MIN should be evaluated in order of ascending value, starting with the lowest-value move.

(c) (3 points) Re-draw the minimax tree for the RED-BLUE game so that, if moves are always evaluated from left to right, the $\alpha - \beta$ search only needs to evaluate 5 of the 8 terminal states.

Solution: The first two leaves should be (1,2) or (2,1). The next two leaves should be (3,4) or (4,3). The last four leaves should be (-1,2,-3,-4), in any order.

Search

Algorithm	Complete?	Optimal?	Time complexity	Space complexity	Implement the Frontier as a…
BFS	Yes	If all step costs are equal	$\mathcal{O}\{b^d\}$	$\mathcal{O}\{b^d\}$	Queue
DFS	No	No	$\mathcal{O}\{b^m\}$	$\mathcal{O}\{bm\}$	Stack
UCS	Yes	Yes	Number of nodes, n, with $g(n) \le C^*$	Number of nodes, n, with $g(n) \le C^*$	Priority Queue sorted by $g(n)$
Greedy	No	No	$\mathcal{O}\{b^m\}$	$\mathcal{O}\{b^m\}$	Priority Queue sorted by h(n)
A*	Yes	Yes	Number of nodes, n, with $f(n) \le C^*$	Number of nodes, n, with $f(n) \le C^*$	Priority Queue sorted by f(n)

b = branching factor, d = length of best path to goal, m = length of longest path to anywhere,
$g(n)$ =cost of best path from start to node n, C^* = cost of best path to goal,
$h(n)$ =heuristic underestimate of best path from node n to goal, $f(n) = g(n) + h(n)$

A* search

- A* search: Using a heuristic to help choose which node to expand
- Proof that Dijkstra's algorithm is optimal:
  - Expanding nodes in order of increasing cost ⇒ first time goal node is expanded it will have the smallest possible cost of any path to goal
- Heuristics that allow A* to be optimal:
  - Consistent: $h(p) \le d(p,r) + h(r)$
  - Admissible: $h(p) \le d(p, Goal)$
- Design a consistent heuristic by relaxing constraints

Minimax

- Alternating two-player zero-sum games
  - $\wedge$ = a max node, $\vee$ = a min node
- Minimax search
  - Max: $v = \max(v, \text{child})$. Min: $v = \min(v, \text{child})$
- Limited-horizon computation and heuristic evaluation functions

$$v(s) = w_1 f_1(s) + w_2 f_2(s) + \cdots$$

- Alpha-beta search
  - Max: $v = \max(v, \text{child})$, $\alpha = \max(\alpha, \text{child})$, prune if $\alpha \ge \beta$.
  - Min: $v = \min(v, \text{child})$, $\beta = \min(\beta, \text{child})$, prune if $\alpha \ge \beta$.
- Computational complexity of minimax and alpha-beta
  - Minimax is $\mathcal{O}\{b^d\}$. With optimal move ordering, alpha-beta is $\mathcal{O}\{b^{d/2}\}$.

AI Safety: Variance Network

Given a dataset of examples $D = \{(x_0, y_0), \ldots, (x_{n-1}, y_{n-1})\}$, the network has two outputs, $f_1(x)$ and $f_2(x)$, trained to minimize:

$$\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n}(f_1(x_i) - y_i)^2 + \frac{1}{n-1}\sum_{i=1}^{n}\left(f_2(x_i) - (f_1(x_i) - y_i)^2\right)^2$$

…learns to estimate the conditional mean and conditional variance:

$$f_1(x_i) \underset{n\to\infty}{\longrightarrow} E[Y|X = x_i]$$
$$f_2(x_i) \underset{n\to\infty}{\longrightarrow} \text{Var}(Y|X = x_i)$$

Forward-chaining

Forward-chaining is a method of proving a theorem, T:

- Starting state: a database of known true propositions, $\mathcal{D} = \{P_1, P_2, \ldots\}$
- Actions: the set of possible actions is defined by a set of rules, where each rule has the form $P \Longrightarrow Q$.
- Neighboring states: if P_1 unifies to P creating $S(P) = S(P_1)$, then create the new database $\mathcal{D}' = \{P_1, P_2, \ldots, S(Q)\}$
- Termination: search terminates when we find a database containing T

Backward-chaining

Backward-chaining is a method of proving a theorem, T:

- Starting state: a goalset containing only one goal, the result to be proven, $\mathcal{G} = \{T\}$
- Actions: the set of possible actions is defined by rules of the form $P_1 \wedge P_2 \wedge \cdots \wedge P_n \Longrightarrow Q$
- Neighboring states: if Q unifies with some $Q' \in \mathcal{G}$ producing $S(Q) = S(Q')$ then:
  - Remove Q' from $\mathcal{G}$
  - Replace it with $S(P_1) \wedge S(P_2) \wedge \cdots \wedge S(P_n)$
- Termination: search terminates if all propositions in the goalset are known to be true.

Vector Semantics

- Skip-Gram:

$$\mathcal{L} = -\frac{1}{T}\sum_{t=0}^{T-1}\sum_{j=-c,j\ne 0}^{c} \ln P(w_{t+j}|w_t)$$

- Continuous Bag of Words (CBOW):

$$\mathcal{L} = -\frac{1}{T}\sum_{t=0}^{T-1}\sum_{j=-c,j\ne 0}^{c} \ln P(w_t|w_{t+j})$$

- Softmax probability, Dot-product similarity:

$$P(W_t = m|W_{t+j} = n) = \frac{\exp(v_m @ v_n)}{\sum_{m'}\exp(v_{m'} @ v_n)}$$

- Train using SGD:

$$v_m \leftarrow v_m - \eta\nabla_{v_m}\mathcal{L} = v_m + \frac{\eta}{T}\sum_{t:w_t=m}\sum_{j=-c,j\ne 0}^{c}\left(1 - P(W_t = m|w_{t+j})\right)v_{w_{t+j}}$$

Transformer

- Stack up v_t, k_t, and q_i into matrices:

$$v = \begin{bmatrix} v_1 \\ \vdots \\ v_n \end{bmatrix}, k = \begin{bmatrix} k_1 \\ \vdots \\ k_n \end{bmatrix}, q = \begin{bmatrix} q_1 \\ \vdots \\ q_m \end{bmatrix}$$

- $\alpha_{i,t}$ is the t[th] output of a softmax whose input vector is $q_i @ k^T$:

$$\alpha_{i,t} = \text{softmax}_t(q_i @ k^T) = \frac{\exp(q_i @ k_t)}{\sum_\tau \exp(q_i @ k_\tau)}$$

- c_i is the product of the vector softmax$(q_i @ k^T)$ times the v matrix:

$$c_i = \text{softmax}(q_i @ k^T) @ v = \sum_t \alpha_{i,t}\, v_t$$