Final Examination

CS 540: Introduction to Artificial Intelligence

May 8, 2006

LAST NAME: _____SOLUTION_____

FIRST NAME: _____

Problem	Score	Max Score
1	_____	10
2	_____	14
3	_____	16
4	_____	19
5	_____	14
6	_____	8
7	_____	19
Total	_____	100

1. [10] **Neural Networks**

(a) [3] Can a Perceptron compute the Majority function, i.e., given binary input values, outputs 1 if there are more 1's in the input than 0's, and outputs 0 otherwise. Either explain why it cannot, or else explain how to set the weights and threshold for the general case where there are *n* input units.

Yes, by setting all weights to 1 and setting the threshold to n/2.

(b) [2] True or False: Training neural networks has the potential problem of over-fitting the training data.

True

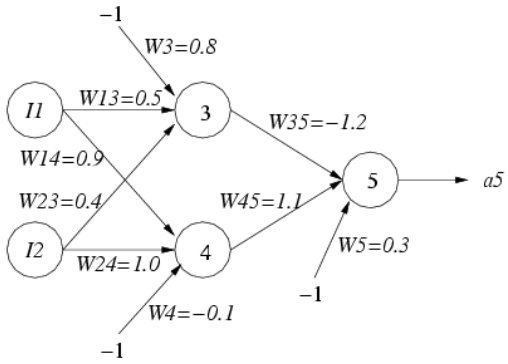
(c) [2] True or False: The Perceptron Learning Rule is a sound and complete method for a Perceptron to learn to correctly classify any two-class problem.

False, it can only learn linearly separable functions.

(e) [3] What is the search space and what is the search method used by the back-propagation algorithm for training neural networks?

The search space is the n-dimensional weight space defined by the n weights in a given network. The search method is gradient descent.

2. [14] **Backpropagation Learning**
Consider the following 2-layer feedforward network (nodes 3, 4, and 5 are sigmoid units):



Given an input example ($I1$=1, $I2$=1) with desired output 0, the above network computes outputs at the two hidden units and one output unit of $a3 = 0.525$, $a4 = 0.88$, and $a5 = 0.51$, respectively.

(a) [8] Compute the following after *one* iteration of backpropagation. Do not use any updated weights in any computation. Use a learning rate parameter value of $\alpha = 0.1$.

(i) [4] $\Delta W35$

The error at the output node is E = (0 - 0.51) = -0.51

$\Delta_5 = a5(1 - a5)E = (0.51)(0.49)(-0.51) = -0.127$

$\Delta W35 = \alpha * a3 * \Delta_5 = (0.1)(0.525)(-0.127) = -0.0067$

(ii) [4] $\Delta W13$

$\Delta_3 = a3(1 - a3) * \Delta_5 * W35 = (.525)(.475)(-.127)(-1.2) = 0.038$

$\Delta W13 = \alpha * I1 * \Delta_3 = (.1)(1)(.038) = 0.0038$

(b) [3] Does the backpropagation algorithm, when run until a minimum is achieved, always find the same solution no matter what the initial set of weights are? Briefly explain why or why not.

No. It will iterate until a local minimum in the squared error is reached.

(c) [3] What is a good way to determine when to stop iterating the backpropagation algorithm?

It is not a good idea to iterate until a local minimum on the training set is achieved because this will often cause over-fitting of the training data. Better is to use a Tuning Set, which is separate from both the Training Set and the Testing Set, to determine when to stop. That is, stop when the Tuning Set's error is minimized while using the Training Set to iteratively update the weights. Performance is then evaluated using the Testing Set.

3. [16] **Probabilistic Reasoning**

1% of women over age forty who are screened, have breast cancer. 80% of women who really do have breast cancer will have a positive mammography (meaning the test indicates she has cancer). 9.6% of women who do *not* actually have breast cancer will have a positive mammography (meaning that they are incorrectly diagnosed with cancer). Define two Boolean random variables, M meaning a positive mammography test and $\neg M$ meaning a negative test, and C meaning the woman has breast cancer and $\neg C$ means she does not.

(a) [6] If a woman in this age group gets a positive mammography, what is the probability that she actually has breast cancer?

Given: P(C) = 0.01, P(M|C) = 0.8, P(M| ¬C) = 0.096.

$$P(C|M) = [P(M|C)P(C)]/P(M)$$

$$= [P(M|C)P(C)]/[P(M|C)P(C) + P(M|\ C)P(\ C)]$$

$$= (0.8)(0.01)/[(0.8)(0.01) + (0.096)(0.99)]$$

$$= (0.008)/(0.008 + 0.09504)$$

$$= 0.0776$$

So, there is a 7.8% chance.

(b) [2] True or False: The "Prior" probability, indicating the percentage of women with breast cancer, is not needed to compute the "Posterior" probability of a woman having breast cancer given a positive mammography.

False, as seen in the use of Bayes's Rule in (a).

(c) [6] Say a woman who gets a positive mammography test, M1, goes back and gets a second mammography, M2, which also is positive. Use the Naive Bayes assumption to compute the probability that she has breast cancer given the results from these two tests.

$$P(C|M1, M2) = [P(M1, M2|C)P(C)]/P(M1, M2)$$

$$= [P(M1|C)P(M2|C)P(C)]/P(M1, M2)$$

$$= (.8)(.8)(.01)/P(M1, M2) = 0.0064/P(M1, M2)$$

Now, if we further assume that M1 and M2 are independent, then P(M1,M2) = P(M1)P(M2) and

$$P(M) = (P(M|C)P(C) + P(M|\neg C)P(\neg C) = (.8)(.01) + (.096)(1-.01) = 0.103$$

Then, P(C|M1,M2) = .0064 / .103 = 0.603 (i.e., 60.3%)

More correctly, we don't assume that M1 and M2 are independent, but only use the original Naive Bayes assumption that M1 and M2 are conditionally indpendent given C. In this case we need to compute P(M1,M2):

$$P(M1, M2) = P(M1, M2|C)P(C) + P(M1, M2|\neg C)P(\neg C)$$

$$= P(M1|C)P(M2|C)P(C) + P(M!|\neg C)P(M2|\neg C)P(\neg C)$$

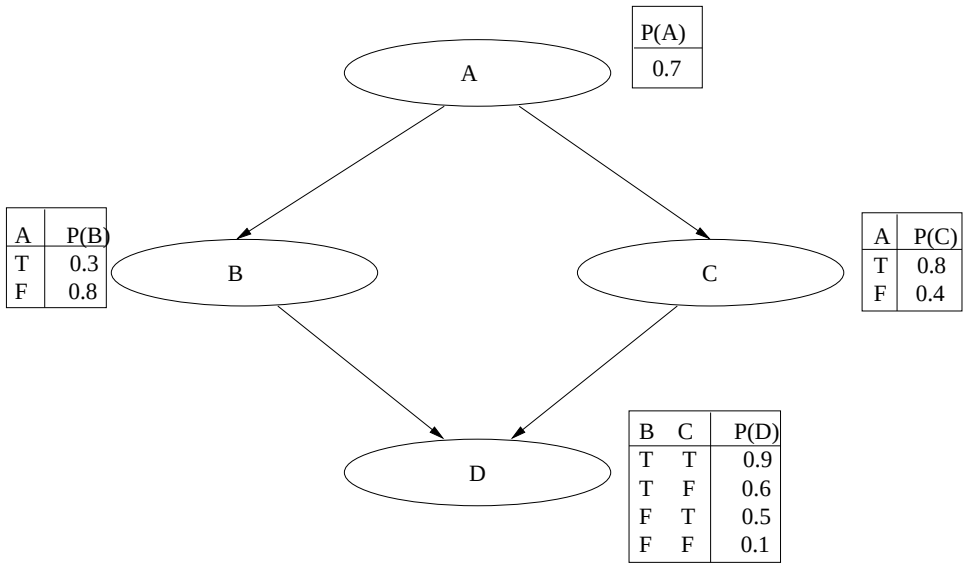
$$= (.8)(.8)(.01) + (.096)(.096)(.99) = 0.0155$$

So, P(C|M1,M2) = 0.603 / 0.0155 = 0.4129 (i.e., 41.3%)

(d) [2] True or False: P(C | M1, M2) can be calculated in general given only P(C) and P(M1, M2 | C).

False. Need either P(M1, M2) or P(M1, M2 | ¬C).

4. [19] **Bayesian Networks**
Consider the following Bayesian Network containing four Boolean random variables:



(a) [2] How many independent values are required to store the full joint probability distribution for this problem?

$2^4 - 1 = 15$

(b) [2] True or False: From the above network it is possible to compute all of the values in the full joint probability distribution.

`True`

(c) [2] True or False: Based on the topology only of the network, P(A,C,D) = P(A)P(C)P(D)

`False`

(d) [2] True or False: Based on the topology only of the network, P(D|C) = P(D|A,C)

`False`

(e) [5] Compute $P(\neg C|A, B)$ where A means $A = true$, $\neg C$ means $C = false$, etc.

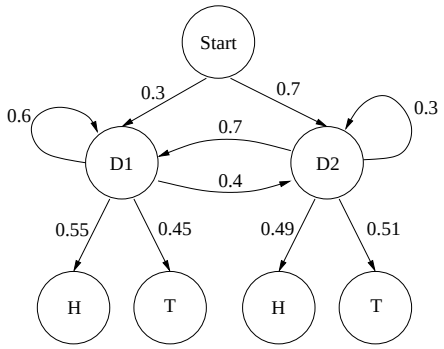
$P(\neg C|A, B) = P(\neg C|A) = 1 - 0.8 = 0.2$

(f) [6] Compute $P(A, B, C, \neg D)$

$$\begin{aligned} P(A, B, C, \neg D) &= P(\neg D|A, B, C)\ P(A, B, C) \\ &= P(\neg D|B, C)\ P(B|A, C)\ P(C|A)\ P(A) \\ &= P(\neg D|B, C)\ P(B|A)\ P(C|A)\ P(A) \\ &= (1 - 0.9)(0.3)(0.8)(0.7) \\ &= 0.0168 \end{aligned}$$

5. [14] **Hidden Markov Models**

I have two coins, D1 and D2. At each turn I pick one of the two coins but don't know which one I picked. Which coin is selected is assumed to obey a first-order Markov assumption. After I pick a coin, I flip it and observe whether it's a Head or a Tail. The two coins are not "fair," meaning that they each don't have equal likelihood of turning up heads or tails. The complete situation, including the probability of which coin I pick first is given by the following HMM, where D1 and D2 are the two hidden states, and H and T are the two possible observable values.



(a) [6] Compute P(T1=D1, T2=D1, T3=D2) meaning at the first turn, T1, coin D1 was picked, etc.

$$P(T1 = D1, T2 = D1, T3 = D2) = P(T3 = D2|T2 = D1)P(T2 = D1|T1 = D1)P(T1 = D1|Start)$$

$= (.4)(.6)(.3)$
$= 0.072$

(b) [6] Compute P(T1=D1 | O1=H) meaning the first turn picked coin D1 given the first observation, O1, was a Head when that coin was flipped.

$$P(T1 = D1|O1 = H) = [P(O1 = H|T1 = D1)P(T1 = D1)]/P(O1 = H)$$

$$= [0.55)(0.3)]/[P(O1 = H|T1 = D1)P(T1 = D1) + P(O1 = H|T1 = D2)P(T1 = D2)]$$

$$= (0.165)/[(0.55)(0.3) + (0.49)(0.7)]$$

$$= 0.165/(0.165 + 0.343)$$

$$= 0.3248$$

(c) [2] Say we generate a sequence of 1,000 turns and their associated observations and the last one was a "T". What is the most likely hidden state corresponding to the last observation?

We must compare which is greater, P(T1000=D1 | O1000=T) or P(T1000=D2 | O1000=T). Using Bayes's Rule and eliminating the denominator, which is the same in both cases, means we must determine the greater of P(O1000=T|T1000=D1)P(T1000=D1) and P(O1000=T|T1000=D2)P(T1000=D2). The first term in each case is

available directly from the HMM. So, what we must compute is P(T1000=D1) and P(T1000=D2). P(T1000=D1) = P(T1000=D1|T999=D1)P(T999=D1) + P(T1000=D1|T999=D2)P(T999=D2) and P(T1000=D2) = P(T1000=D2|T999=D1)P(T999=D1) + P(T1000=D2|T999=D2)P(T999=D2). In steady state, P(T1000=D1) = P(T999=D1) = (1 - P(T1000=D2)) = (1 - P(T999=D2)). Solving, we get P(D1) = 0.636. So, we get for D1: (.45)(.636) = 0.286 and for D2: (.51)(1 - .636) = 0.186. D1 > D2 so D1 is the most likely state.

6. [8] **Face Recognition using Eigenfaces and SVM**

(a) [2] What classification method is used by the Eigenface algorithm to recognize the face in a test image? That is, given the representations of the 100 people in face space and the representation of the test face, what method is used to classify the test face image?

Nearest neighbor in 10D weight space.

(b) [6] Give one major advantage and one major disadvantage of using a Support Vector Machine (SVM) to recognize 100 different people's faces compared to the Eigenface method.

Advantages: Given a set of training images of a single person, a non-linear boundary can be derived (using a kernel function) for separating this person from the other people's faces. It maximizes the margin between classes, which gives better robustness. There are relatively efficient implementations relative to Eigenface analysis to define a good subspace.

Disadvantages: SVM only allows 2-class decisions, so one would need 100 SVMs, one each for comparing one person against all others. SVM requires some kind of pre-processing to extract an appropriate feature vector from each image since using the entire image is not realistic or robust. Also, hard to distinguish faces from non-faces.

7. [19] **Planning**
Consider a robot with the following three STRIPS operators:

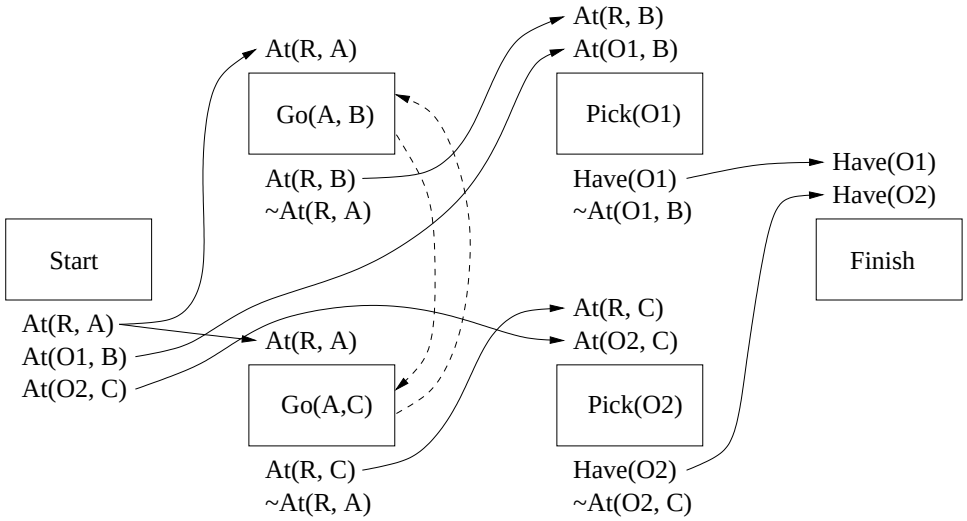
Operator	Preconditions	Add	Delete
GO(x,y)	At(Robot,x)	At(Robot,y)	At(Robot,x)
PICK(o)	At(Robot,x), At(o,x)	Have(o)	At(o,x)
DROP(o)	At(Robot,x), Have(o)	At(o,x)	Have(o)

(a) [3] Write an *effect axiom* in the Situation Calculus that captures the same meaning as the operator $PICK(o)$.

$$\forall x, o, s\ [At(Robot, x, s) \langle At(o, x, s)] \Rightarrow [Have(o, Result(PICK(o), s)) \langle \neg At(o, x, Result(PICK(o), s))]$$

(b) [16] Using the given STRIPS operators, I want to use the POP algorithm for partial-order planning. Assume the initial state is: At(Robot,A) and At(O1,B) and At(O2,C), and the goal state is: Have(O1) and Have(O2). Assume I've generated a partial plan that consists of Start and Finish states plus states for 4 steps: Go(A,B), Pick(O1), Go(A,C), and Pick(O2).

(i) [10] Draw the partial plan associated with this set of 6 states and include all causal links showing there are no open preconditions. Draw the states as boxes with preconditions listed on top of the box and effects below the box.



(ii) [3] What threat(s) is/are in your partial plan drawn in (i)?

There are 2 threats. Step Go(A, C) has an effect, $\neg At(R, A)$, so this step threatens the causal link from Start to Go(A, B) that solves sub-goal At(R, A). Similarly, step Go(A, B) threatens the causal link from Start to Go(A, C) that solves sub-goal At(R, A).

(iii) [3] Add one or more temporal links to (i) that remove all threats.

To remove the first threat, add a temporal link from Go(A, B) to Go(A, C) which promotes the threatening step Go(A, C). Similarly, to remove the second threat, add a temporal link from Go(A, C) to Go(A, B) which promotes the threatening step Go(A, B). Note that this introduces a cycle into the graph, which is impossible, so this would initiate backtracking to find a different way to solve the open preconditions.