Number:

Problem	Points	Score
1 (a)	10	
1 (b)	15	
1 (c)	15	
1 (d)	15	
1 (e)	15	
2 (a)	20	
2 (b)	10	
Total	100	

Notes:

1. The exam is closed books and notes. You are allowed one 8 1/2" x 11" double-sided sheet of notes.

2. Please indicate clearly your answer to the problem by some form of highlighting (underlining).

3. Your solutions must be legible and easy to follow. If I can't read it or understand it, it is wrong. Random scribbling will not receive credit.

4. Please show ALL work. Answers with no supporting explanations or work will be given no credit.

5. Several problems on this exam are fairly open-ended. Since the evaluation of your answers is obviously a subjective process, we will use a market place strategy in determining the grade. Papers will be rank-ordered in terms of the quality of the solutions, and grades distributed accordingly.

1. Before his death, Chairman Mao devised a plot to overthrow the western world by designing a coin called the "Jonnie," shown to the right. This coin had three sides: "Top" denoted "T", "Bottom" denoted "B," and "Sides", denoted "S". The plot involved playing the ancient Chinese game known as JonGo, and betting on the outcome. Chairman Mao demonstrated this game to his top military advisors by flipping the coin 10 consecutive times, and doing three separate trials. He generated the following observations:

Trial No. 1: BBBTTBBBTT
Trial No. 2: TTBBBTTBBB
Trial No. 3: BBBBBBTTTS

Your job is to model this system using a speech recognition system. Let's begin with some simple calculations.

(a) Estimate a unigram probability density function for the observable symbols in this system (the state of the coin after each toss). Then compute the entropy: $H(x) = -\sum p(x)\log_2 p(x)$.

There are three symbols:

P(B) = 18/30 = 0.6000

P(T) = 11/30 = 0.3667

P(S) = 1/30 = 0.0333

The entropy is:

$$H(x) = -\sum p(x)\log_2 p(x) = -\sum 0.6\log_2 0.6 + 0.3667\log_2 0.3667 + 0.0333\log_2 0.0333$$
$$= -\sum 0.6\log_2 0.6 + 0.3667\log_2 0.3667 + 0.0333\log_2 0.0333$$
$$= 1.1364 \text{ bits}$$

(b) Estimate a bigram probability density function for this coin. (**Do not take into account an end of string symbol**.) Compute the entropy. Compare and contrast this number with the answer to part (a). Why are they different?

Let's consider the joint probability distribution, $p(w_1,w_2)$, and treat the trials as three separate sequences (enumerate all two symbol sequences that have occurred). This implies there are 27 possible bigrams, ignoring edge effects:

P(B,B) = 13/27	P(B,T) = 3/27	P(B,S) = 0
P(T,B) = 4/27	P(T,T) = 6/27	P(T,S) = 0
P(S,B) = 0	P(S,T) = 1/27	P(S,S) = 0

The entropy of this distribution is: $H(x) = 1.93\text{bits}$. Note that this number is less than twice the entropy computed in (a). This is because not all symbols are equally likely to follow a given symbol (for example, S only follows T). Hence, we can exploit these constraints to better predict the outcome of seemingly random coin tosses.

(c) Smooth the bigram probability distribution using any two smoothing techniques discussed in class. Compare and contrast the entropies computed for these smoothed distributions. Why are they different?

(1) Simple smoothing: add 1 to each count

P(B,B) = 14/36 P(B,T) = 4/36 P(B,S) = 1/36

P(T,B) = 5/36 P(T,T) = 7/36 P(T,S) = 1/36

P(S,B) = 1/36 P(S,T) = 2/36 P(S,S) = 1/36

The resulting entropy is: $H(x) = 2.54$ bits.

(2) To keep this simple, let's use discounting. We have four non-zero entries, so we need to recover at least 4 counts. Let's set D=1:

P(B,B) = 12/27 P(B,T) = 2/27 P(B,S) = 1/27

P(T,B) = 3/27 P(T,T) = 5/27 P(T,S) = 1/27

P(S,B) = 1/27 P(S,T) = 1/27 P(S,S) = 1/27

The resulting entropy is: $H(x) = 2.17$ bits.

Smoothing broadens the distribution, which must increase entropy as compared to our answer in (b), because we are allowing more possibilities (in this case, allowing symbols that did not occur in the training data to have some non-zero probability). Simple smoothing broadens the distribution more than discounting in this case, thought that is dependent on the discounting threshold.

(d) Describe a language model that will better model the data for this experiment. Justify your answer using as many detailed calculations as possible.

I would propose the use of a trigram LM. Very few trigrams appear in the training data:

BBB (8), BBT (3), BTT (3), TTB (2), TBB (2), TTT (1), TTS (1)

The entropy is: $H(x) = 2.25$ bits. This is much lower than $NH(x)$ for the unigram case. This means there is redundancy in the language that we can exploit via a language model.

On the other hand, training a trigram LM on such small data could be dangerous :)

(e) Compute the training set perplexity using the unigram language model of (a). Compare and contrast this value with the average branching factor. Explain how your observations might be relevant to the speech recognition problem.

The training set perplexity and the entropy of the unigram language model of (a) are essentially the same thing:

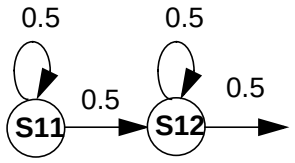
$PP = 2^{H(x)} = 2.19$ Avg. Branching Factor = 3

Since the symbol "S" only appears once, it is easy to understand that the perplexity is much less than the average branching factor — the coin really uses only 2 symbols :)

2. Some smart American government employees, who work for an agency that doesn't exist, decide they can sabotage this plot by training a speech recognition system on this data. The system they develop looks like this:

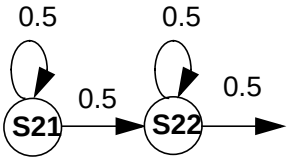
Language Model:	Lexicon:	Acoustic Models:
P(T) = 0.50	T -> S1 S2	
P(B) = 0.25	B -> S2 S1	
P(S) = 0.25	S -> S2 S2	

Model S1:



P(T) = 0.75	P(T) = 0.25
P(B) = 0.25	P(B) = 0.75
P(S) = 0.00	P(S) = 0.00

Model S2:

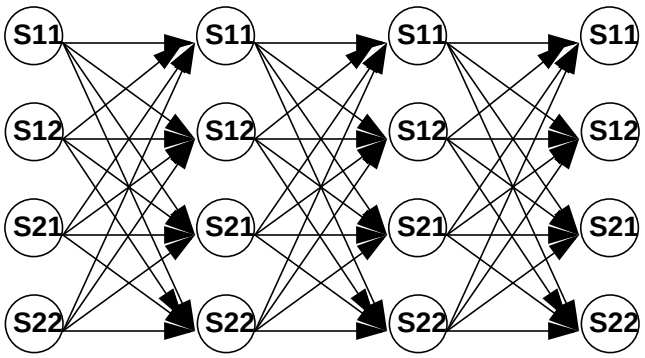


P(T) = 0.25	P(T) = 0.25
P(B) = 0.25	P(B) = 0.50
P(S) = 0.50	P(S) = 0.25

(a) Find the model and state sequences that were most likely to produce the observation sequence "TB".

This problem was easier than I had anticipated. The lexicon requires a sequence of two models. Each model, shown to the right, consumes two frames of data. Hence, the only valid output sequences these models can produce must consume four frames of data.

Originally, I was going to specify four frames of observations. In this case, the approach I would have taken was to set up a four frame dynamic programming:



I would then consider only those transitions that were allowed by the lexicon (for example S1 can only transition to S2 for the symbol T, and S2 can only go to S1 for the symbol B). Then I would apply the language model weights accordingly (because you know the word history for each path). This was an attempt to get you to step through a time synchronous decoding.

For the stated observation sequence "TB", state sequence S11 S12 was the most likely, which means it is likely that T was the first output symbol observed (according to the lexicon).

(b) Estimate the computational complexity of your calculation in part (a) and discuss how you might make it more efficient.

The complexity of the time synchronous search shown above is $O(N^2T)$. This is too complicated for LVCSR. We must use beam search techniques to reduce the number of active predecessor states so that the complexity becomes O(NT).