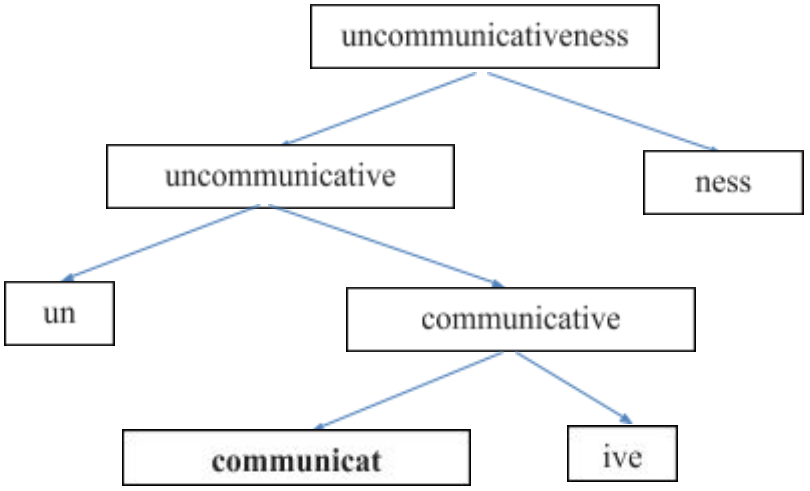
CS626	**Speech, Natural Language Processing and the Web**	**Midsem**
5/10/20	**Time- 9.30am to 10.30am**	**marks-30**

(All questions are multiple choice questions. Some questions- if the answer is wrong- carry negative marks; ***they are marked * and also have the expression (negative) beside the marks***. For all questions, think carefully and work out the solution before committing to the option you think is correct.)

Q1. "Stem" is the root or the main part of a word, to which morphemes (inflections and other formative elements) are added. Stem is also a morpheme in its own right. Note that there may be phonological changes at the boundaries of morphemes leading to changes of graphemes (e.g., lady+s=ladies; gate+ing=gating).

The length of the stem in the word "uncommunicativeness" is

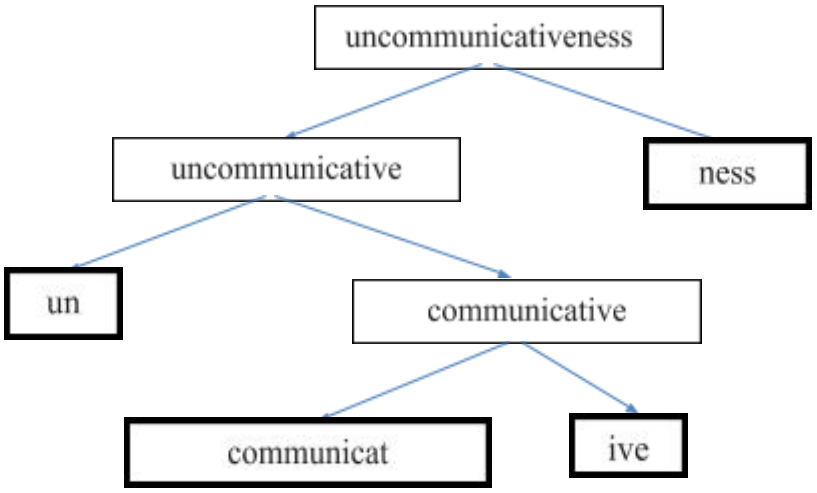
(a) 10
(b) 12
(c) 13
(d) None of the above marks: 2



The length of the stem 'communicat' is 10. Hence, the answer is (a) 10

Q2. The number of the morphemes in the word “uncommunicativeness” is:

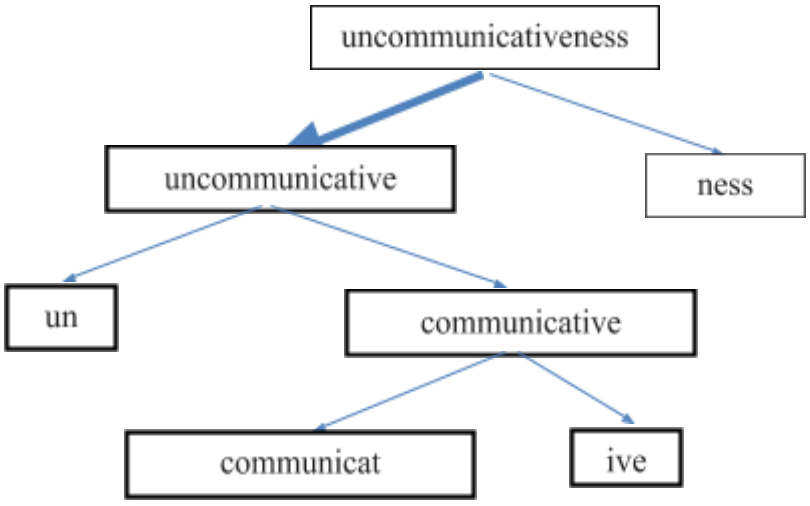
(a) 3
(b) 4
(c) 2
(d) None of the above marks: 3



The number of leaf nodes is the number of morphemes, *i.e.,* 4. Hence, the correct option is (b) 4

Q3. The process of constructing a tree of substrings from a word, with the given word being the root and the morphemes being the leaves is called morphological parsing. The morphological parse tree for the word “uncommunicativeness” is

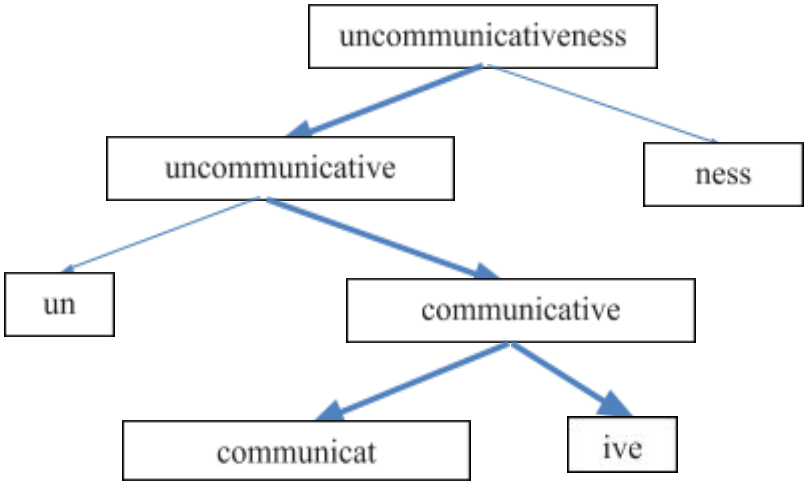
(a) Left-skewed (i.e., more nodes on the left subtree of the root)
(b) Right-skewed
(c) Balanced
(d) None of the above marks: 3



The Tree can be clearly seen to be left-skewed. Hence the correct option is (a) Left-skewed (*i.e.,* more nodes on the left subtree of the root)

Q4*. The maximum depth of the parse tree (i.e., the length of path from the root to a leaf) of the word "uncommunicativeness" is

(a) 2
(b) 4
(c) 3
(d) 5 marks: 3 (negative 1)



The maximum depth of the tree is 3, given either of the longest paths. Hence, the correct answer is (c) 3

Q5*. I have purchased a sentiment analysis software. Given a sentence as input, the s/w outputs 1 or 0, depending on whether the sentiment expressed in the sentence is positive or negative. Thus given "this movie is really good", the output is 1. But for "the songs are pretty ordinary", the output is 0. One day, the domain where I have to use the software, changes from "movie" to "finance" where I have to deal with reports like "the market is upbeat" (positive sentiment), "the customers of the company are worried" (negative). I ask the supplier to "update" the s/w and give me the updated version. The Machine Learning engineer of the company interacts with me and asks me to give him values of "what is the total number of times 'upbeat' appears in all positive sentiment sentences as against in all negative sentences" and similar such statistics. What kind of base ML model the engineer is *most likely* using for training the sentiment analyser:

(a) Discriminative
(b) Generative
(c) Can be either
(d) None of the above 3 marks (negative 1)

(Answer on Next Page)

Answer with explanation:

The key sentence in the question is, "what is the total number of times 'upbeat' appears in all positive sentiment sentences as against in all negative sentences".

This means that we need the ratio of how many times 'upbeat' appears in positive sentiment sentences to the total number of times 'upbeat' appears in both positive and negative sentences. This is the case for all other statistics.

Here we are interested in *p(upbeat|positive sentiment)* and similar probability values. Thus, Bayes' Theorem has been applied to the probability of interest *p(pos. or neg. | upbeat and such other features)*. **This clearly shows that the model is Generative.**

Q6*. In the class we have described a Markov Model that 'decodes' (i.e., tags) from left to right. If you were to decode from right to left, the result compared to 'left to right" will be:

(a) Worse
(b) Better
(c) Equivalent
(d) None of the above.

Assume, (i) bigrams for state transitions (i.e., $P(t_i|t_{i-1})$) and (ii) that the probabilities of first state (sentence beginner) and last states (sentence finisher) are the same.

4 marks (negative 1)

Ans:

L->R and R->L solutions will be the same. Lexical probabilities are insensitive to the direction of scan. We examine the transition probabiltiles.

Proof :

(bigram assumption) :

L->R

$$P(T) = P(T_0 = t_0 = \hat{}) \prod_{i=1}^{n+1} P(T_i = t_i \mid T_{i-1} = t_{i-1})$$

$$= P(T_0 = \hat{}) \frac{\prod_{i=1}^{n+1} P(T_i = t_i, T_{i-1} = t_{i-1})}{P(T_0 = \hat{}) \prod_{i=1}^{n} P(T_i = t)} = \frac{\prod_{i=1}^{n+1} P(T_i = t_i, T_{i-1} = t_{i-1})}{\prod_{i=1}^{n} P(T_i = t)}$$

R->L

$$P(T) = P(T_{n+1} = t_{n+1} = \$) \prod_{i=n+1}^{1} P(T_{i-1} = t_{i-1} \mid T_i = t_i)$$

$$= P(T_{n+1} = \$) \frac{\prod_{i=n+1}^{1} P(T_{i-1} = t_{i-1}, T_i = t_i)}{P(T_{n+1} = \$) \prod_{i=n}^{1} P(T_i = t_i)} = \frac{\prod_{i=n+1}^{1} P(T_{i-1} = t_{i-1}, T_i = t_i)}{\prod_{i=n}^{1} P(T_i = t_i)}$$

Both the expressions are the same.

(trigram assumption)

L->R

$$P(T) = P(T_0 = t_0 = \hat{}) P(T_1 = t_1 \mid T_0 = t_0 = \hat{}) \prod_{i=2}^{n+1} P(T_i = t_i \mid T_{i-1} = t_{i-1}, T_{i-2} = t_{i-2})$$

$$= P(T_0 = t_0 = \hat{}) \frac{P(T_1 = t_1, T_0 = t_0 = \hat{})}{P(T_0 = t_0 = \hat{})} \frac{\prod_{i=2}^{n+1} P(T_i = t_i, T_{i-1} = t_{i-1}, T_{i-2} = t_{i-2})}{\prod_{j=1}^{n} P(T_i = t_i, T_{i-1} = t_{i-1})}$$

$$= \frac{\prod_{i=2}^{n+1} P(T_i = t_i, T_{i-1} = t_{i-1}, T_{i-2} = t_{i-2})}{\prod_{j=2}^{n} P(T_i = t_i, T_{i-1} = t_{i-1})}$$

R->L

$$P(T) = P(T_{n+1} = t_{n+1} = \$) P(T_n = t_n \mid T_{n+1} = t_{n+1} = \$) \prod_{i=n+1}^{2} P(T_{i-2} = t_{i-2} \mid T_{i-1} = t_{i-1}, T_i = t_i)$$

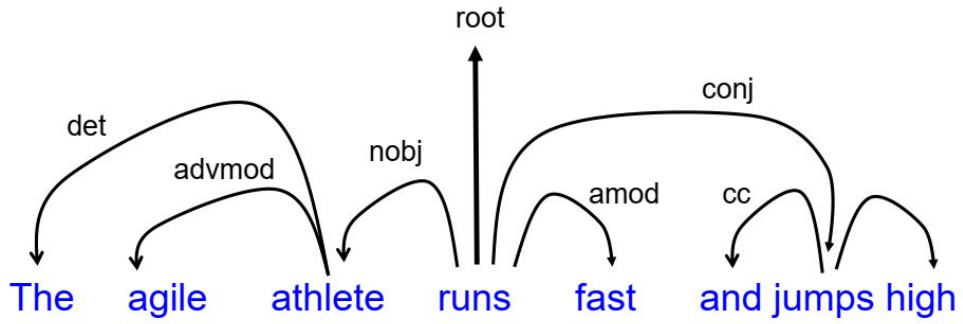
$$= P(T_{n+1} = t_{n+1} = \$) \frac{P(T_n = t_n, T_{n+1} = t_{n+1} = \$)}{P(T_{n+1} = t_{n+1} = \$)} \frac{\prod_{i=n+1}^{2} P(T_{i-2} = t_{i-2}, T_{i-1} = t_{i-1}, T_i = t_i)}{\prod_{i=n+1}^{2} P(T_{i-1} = t_{i-1}, T_i = t_i)}$$

$$= \frac{\prod_{i=n+1}^{2} P(T_{i-2} = t_{i-2}, T_{i-1} = t_{i-1}, T_i = t_i)}{\prod_{i=n}^{2} P(T_{i-1} = t_{i-1}, T_i = t_i)}$$

Again both expressions are the same.

The above method can be generalized to *k*-order HMM for any *k*.

Q7. How many things are **wrong** in the following dependency parse tree. You have to give the exact number, not less and not more.



(a) 3
(b) 5
(c) 6
(d) 0 4 marks

There are five mistakes as follows:

1. Direction of arrow for 'root' is wrong. The arrow goes from head to modifier. In dependency parsing, root is the head for the overall sentence. Overall sentence is the modifier for the root.
2. 'athlete' to 'agile' – 'advmod' is wrong. It should be 'amod' because 'advmod' is adverbial modifier and 'amod' is adjectival modifier.
3. 'athlete' is running and is the subject. Hence, the label from 'runs' to 'athlete' should be 'nsubj'. 'nobj' means nominal object.
4. 'runs' to 'fast' should be 'advmod' and NOT 'amod'.
5. 'jumps' to 'high' – there is no label present.

Hence, the correct answer is (b) 5.

Q8*. Consider the probabilistic context free grammar given below. *'hi'* is the terminal string.

S -> hi; 1/3

S -> S S; 2/3

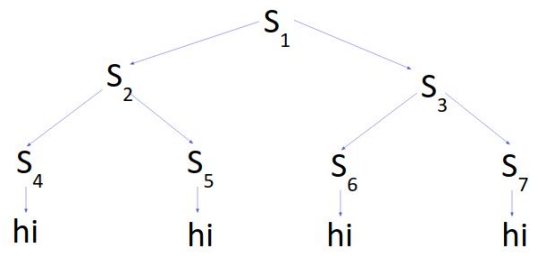
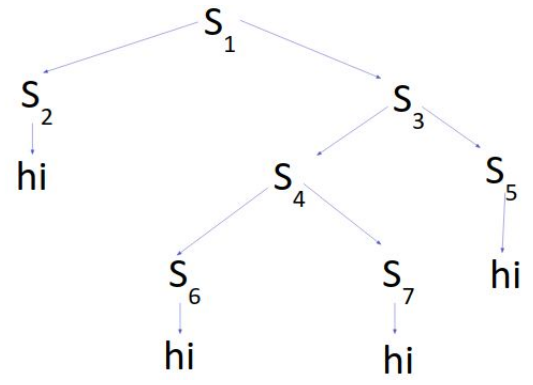
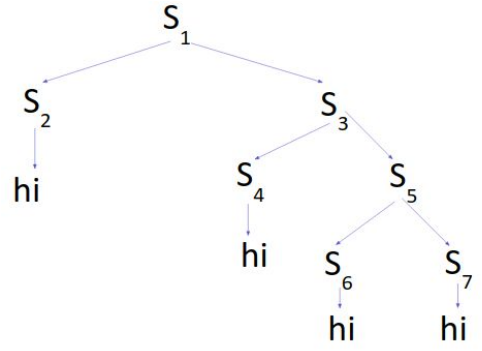
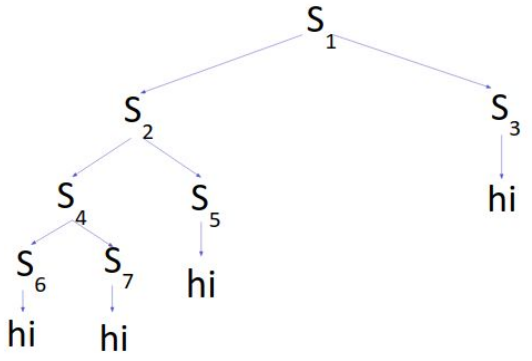
The probability values are beside the rules. The number of parse trees the string *"hi hi hi hi"* has is:

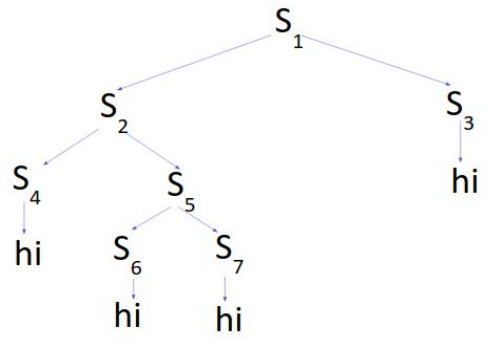
(a) 3
(b) 5
(c) 4
(d) 8 4 marks (negative 1)

(Answer with Parse trees drawn on the next few pages)

Ans:

There are total 5 parse trees

Parse tree 1 	Parse tree2 
Parse tree 3 	Parse tree 4 
Parse tree 5	



Q9*. Consider the probabilistic context free grammar:

S -> hi; 1/3

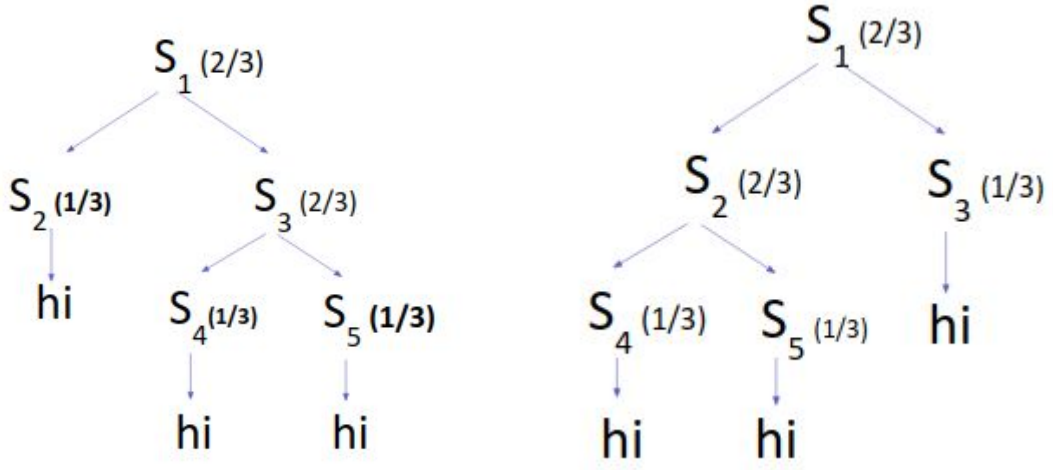
S -> S S; 2/3

The probability of the string "hi hi hi" is:

(a) $2^4/3^5$
(b) $2^5/3^5$
(c) $2^3/3^5$
(d) $2^2/3^5$ 4 marks (negative 1)

Ans:

There are two parse tree possible:



- Probability of the parse tree 1 = 2/3 * 1/3 *2/3 * 1/3 * 1/3 = $2^2 / 3^5$
- Probability of the parse tree = 2/3 * 1/3 *2/3 * 1/3 * 1/3 = $2^2 / 3^5$
- The final probability of the sentence is the sum of the probabilities of its parse trees.
- Hence, the correct answer is $(2^2 / 3^5) + (2^2 / 3^5) = 2^3 / 3^5$

=====Paper Ends=====