CMPT 413 - Spring 2011 - Midterm #2

Please write down "Midterm #2" on the top of the answer booklet.

When you have finished, return your answer booklet along with this question booklet.

(1) The probability model $P(t_i \mid t_{i-2}, t_{i-1})$ is provided below where each t_i is a part of speech tag, e.g. $P(\mathrm{D} \mid \mathrm{N}, \mathrm{V}) = \frac{1}{3}$. Also provided is $P(w_i \mid t_i)$ that a word w_i has a part of speech tag t_i, e.g. $P(\text{flies} \mid \mathrm{V}) = \frac{1}{2}$.

The part of speech tag definitions are: `bos` (*begin sentence marker*), N (*noun*), V (*verb*), D (*determiner*), P (*preposition*), `eos` (*end of sentence marker*).

$P(t_i \mid t_{i-2}, t_{i-1})$	t_{i-2}	t_{i-1}	t_i
1	bos	bos	N
$\frac{1}{2}$	bos	N	N
$\frac{1}{2}$	bos	N	V
$\frac{1}{2}$	N	N	V
$\frac{1}{2}$	N	N	P
$\frac{1}{3}$	N	V	D
$\frac{1}{3}$	N	V	V
$\frac{1}{3}$	N	V	P
1	V	D	N
1	V	V	D
1	N	P	D
1	V	P	D
1	P	D	N
1	D	N	eos

$P(w_i \mid t_i)$	t_i	w_i
1	D	an
$\frac{2}{5}$	N	time
$\frac{2}{5}$	N	arrow
$\frac{1}{5}$	N	flies
1	P	like
$\frac{1}{2}$	V	like
$\frac{1}{2}$	V	flies
1	eos	eos
1	bos	bos

a. Consider a Jelinek-Mercer style interpolation smoothing scheme for $P(w_i \mid t_i)$:

$$P_{jm}(w_i \mid t_i) = \Lambda[t_i] \cdot P(w_i \mid t_i) + (1 - \Lambda[t_i]) \cdot P(w_i)$$

Λ is an array with a value $\Lambda[t_i]$ for each part of speech tag t_i, such that $0 \le \Lambda[t_i] \le 1$. Provide a condition on Λ that must be satisfied to ensure that P_{jm} is a well-defined probability model.

Answer: Because of the following fact about $P(w_i \mid t_i)$:

$$\sum_{w_i} P(w_i \mid t_i) = 1$$

and in P_{jm} we are given t_i, so to interpolate with $P(w_i)$ the following condition has to hold:

$$\sum_{t_i} \Lambda[t_i] = 1$$

b. Provide a Hidden Markov Model (*hmm*) that uses the trigram part of speech probability $P(t_i \mid t_{i-2}, t_{i-1})$ as the transition probability $P_{hmm}(s_j \mid s_k)$ and the probability $P(w_i \mid t_i)$ as the emission probability $P_{hmm}(w_j \mid s_j)$.

Important: Provide the *hmm* in the form of two tables as shown below. The first table contains transitions between states in the *hmm* and the transition probabilities and the second table contains the words emitted at each state and the emission probabilities. Do not provide entries with zero probability.

from-state s_k	to-state s_j	$P(s_j \mid s_k)$

state s_j	emission w	$P(w \mid s_j)$

Hint: In your *hmm* the state $\langle N, \texttt{eos}\rangle$ will have emission of word `eos` with probability 1 and will not have transitions to any other states.

Answer: Here are the two tables that define the HMM, the transition table on the left and the emission table on the right:

from-state s_k	to-state s_j	$P(s_j \mid s_k)$	
bos, bos	bos, N	$P(N \mid bos, bos)$	1
bos, N	N, N	$P(N \mid bos, N)$	$\frac{1}{2}$
bos, N	N, V	$P(V \mid bos, N)$	$\frac{1}{2}$
N, N	N, V	$P(V \mid N, N)$	$\frac{1}{2}$
N, N	N, P	$P(P \mid N, N)$	$\frac{1}{2}$
N, V	V, D	$P(D \mid N, V)$	$\frac{1}{3}$
N, V	V, V	$P(V \mid N, V)$	$\frac{1}{3}$
N, V	V, P	$P(P \mid N, V)$	$\frac{1}{3}$
V, D	D, N	$P(N \mid V, D)$	1
V, V	V, D	$P(D \mid V, V)$	1
N, P	P, D	$P(D \mid N, P)$	1
V, P	P, D	$P(D \mid V, P)$	1
P, D	D, N	$P(N \mid P, D)$	1
D, N	N, eos	$P(eos \mid D, N)$	1

state s_j	emission w	$P(w \mid s_j)$
bos, bos	bos	1
bos, N	time	$\frac{2}{5}$
bos, N	arrow	$\frac{2}{5}$
bos, N	flies	$\frac{1}{5}$
N, N	time	$\frac{2}{5}$
N, N	arrow	$\frac{2}{5}$
N, N	flies	$\frac{1}{5}$
N, V	like	$\frac{1}{2}$
N, V	flies	$\frac{1}{2}$
V, D	an	1
V, V	like	$\frac{1}{2}$
V, V	flies	$\frac{1}{2}$
N, P	like	1
V, P	like	1
P, D	an	1
D, N	time	$\frac{2}{5}$
D, N	arrow	$\frac{2}{5}$
D, N	flies	$\frac{1}{5}$

c. Based on your *hmm* constructed in 1b. what is the state sequence that would be provided by the Viterbi algorithm for the following input sentence:

`bos bos` time flies like an arrow `eos`

Answer:
Note that the only ambiguous words are *flies* (could be N or V) and *like* (could be V or P) and so all you need to do is compare the scores for the following sub-sequence. The bold-faced outcome wins for this sub-sequence which determines the best state sequence for the entire input.

flies	like	
$\mathbf{(N, V)}$	$\mathbf{(V, P)}$	$\mathbf{\frac{1}{2} \times \frac{1}{3} \times 1}$
(N, V)	(V, V)	$\frac{1}{2} \times \frac{1}{3} \times \frac{1}{2}$
(N, N)	(N, P)	$\frac{1}{5} \times \frac{1}{2} \times 1$
(N, N)	(N, V)	$\frac{1}{5} \times \frac{1}{2} \times \frac{1}{2}$

Since the best state sequence is then (bos,N)–(N,V)–(V,P)–(P,D)–(D,N)–(N,eos) the output best state sequence will be *bos/bos, time/N, flies/V, like/P, an/D, arrow/N, eos/eos*.

Answer: The full table is given below but you do not need to compute the entire table to solve this question.

bos	time	flies	like	an	arrow	eos	
(bos,bos) 1	(bos,N) $\times 1 \times \frac{2}{5}$	(N,V) $\times \frac{1}{2} \times \frac{1}{2}$	(V,P) $\times \frac{1}{3} \times 1$	(P,D) $\times 1 \times 1$	(D,N) $\times 1 \times \frac{2}{5}$	(N,eos) $\times 1 \times 1$	$= \frac{1}{75}$
(bos,bos) 1	(bos,N) $\times 1 \times \frac{2}{5}$	(N,V) $\times \frac{1}{2} \times \frac{1}{2}$	(V,V) $\times \frac{1}{3} \times \frac{1}{2}$	(V,D) $\times 1 \times 1$	(D,N) $\times 1 \times \frac{2}{5}$	(N,eos) $\times 1 \times 1$	$= \frac{1}{150}$
(bos,bos) 1	(bos,N) $\times 1 \times \frac{2}{5}$	(N,N) $\times \frac{1}{2} \times \frac{1}{5}$	(N,P) $\times \frac{1}{2} \times 1$	(P,D) $\times 1 \times 1$	(D,N) $\times 1 \times \frac{2}{5}$	(N,eos) $\times 1 \times 1$	$= \frac{1}{125}$
(bos,bos) 1	(bos,N) $\times 1 \times \frac{2}{5}$	(N,N) $\times \frac{1}{2} \times \frac{1}{5}$	(N,V) $\times \frac{1}{2} \times \frac{1}{2}$	(V,D) $\times \frac{1}{3} \times 1$	(D,N) $\times 1 \times \frac{2}{5}$	(N,eos) $\times 1 \times 1$	$= \frac{1}{750}$

(2) **Context-free Grammars**:
For the CFG G given below:

$$\begin{aligned} S &\rightarrow A \mid c \\ A &\rightarrow B\ a \\ B &\rightarrow b\ S \end{aligned}$$

a. What is the language $L(G)$?

Answer: $b^n c a^n : n \geq 0$

b. What is the tree set $T(G)$? You can use any convenient short-hand notation to represent an infinite set of trees.

Answer:

```
T(G) = {
(S (A (B b
         (S (A (B b
                  (S ... c))
               a))
      a)))
}
```

or:

```
T(G) =
{ (S c) ,
  (S (A (B b (S c)) a)),
  (S (A (B b (S (A (B b (S c)) a)) a))),
  ...
}
```

c. Assign probabilities to each rule in the CFG above so that for each string $w \in L(G)$:

$$P(w) = exp\left(\frac{|w| - 1}{2} * ln(0.3) + ln(0.7)\right)$$

where, $|w|$ is the length of string w, *exp* is exponentiation, and *ln* is *log* base e.

Answer:
```
0.3 S -> A
0.7 S -> c
1.0 A -> B a
1.0 B -> b S
```

d. Convert the PCFG from the answer to Q (2c) into Chomsky Normal Form (CNF). The CNF grammar must also have the right probabilities.

Answer:
```
0.7 S -> c
0.3 S -> B A'
1.0 B -> B' S
1.0 B' -> b
1.0 A' -> a
```

e. Provide a leftmost derivation using your CNF grammar for the input string *bbcaa*. The derivation should include probabilities for each step – you can keep them as multiples of individual probabilities if you wish.

Answer:
```
S => BA' (0.3)
=> B'SA' (0.3 * 1.0)
=> bBA'A' (0.3)
=> bB'SA'A' (0.3 * 0.3)
=> bbSA'A' (0.09)
=> bbcA'A' (0.09 * 0.7)
=> bbcaA' (0.027)
=> bbcaa (0.027)
```

f. Briefly explain why conversion into CNF implies that we can parse any CFG in time $O(G^2 n^3)$ where G is the number of non-terminals in the CFG, and n is the length of the input string. (A short 1–2 line answer will suffice).

Answer: The rules of type $A \rightarrow a$ can be used to fill in each span of length one, and to find a start symbol spanning the entire string we use rules of type $A \rightarrow BC$ to recursively find a span i, j if we have previously found a span from i, k for B and k, j for C. Since B and C span over all non-terminals we need G^2 time, and since we need to find all spans i, j with every k in between we need n^3 time.