Midterm, COMS 4705

Name:

Uni:

20	8	18	10	10	10

Good luck!

Part #1 — 20 points

Question 1 (5 points) We define a PCFG where non-terminal symbols are $\{S, A, B\}$, the terminal symbols are $\{a, b\}$, and the start non-terminal (the non-terminal always at the root of the tree) is S. The PCFG has the following rules:

S → A A	0.6
S → A B	0.4
A → A B	0.7
A → a	0.2
A → b	0.1
B → A B	0.9
B → a	0.05
B → b	0.05

For the input string *aab* show two possible parse trees under this PCFG, and show how to calculate their probability.

Question 2 (8 points) Recall that a PCFG defines a distribution $p(t)$ over parse trees t. For any sentence s, if we define $\mathcal{T}(s)$ to be the set of valid parse trees for the sentence s, the probability of the sentence under the PCFG is

$$p(s) = \sum_{t \in \mathcal{T}(s)} p(t)$$

Recall also that for a bigram HMM the probability of any sentence $x_1 \ldots x_n$ under the HMM is

$$p(x_1 \ldots x_n) = \sum_{y_1 \ldots y_{n+1}} p(x_1 \ldots x_n, y_1 \ldots y_{n+1})$$

where the sum is over all state sequences with $y_{n+1} = \text{STOP}$, and $p(x_1 \ldots x_n, y_1 \ldots y_{n+1})$ is the joint probability of the pair of sequences $x_1 \ldots x_n$ and $y_1 \ldots y_{n+1}$ under the HMM.

Now consider the following PCFG:

S $\rightarrow$ NP VP	1.0
VP $\rightarrow$ V NP	1.0
NP $\rightarrow$ John	0.6
NP $\rightarrow$ Mary	0.4
V $\rightarrow$ saw	1.0

In the space below, write down an HMM that gives the same distribution over sentences as the PCFG shown above. You should write down: 1) the set $\mathcal{K}$ of states in the HMM; 2) the set $\mathcal{V}$ of words in the HMM; 3) the transition parameters in the HMM; 4) the emission parameters in the HMM.

(Page intentionally left blank.)

Question 3 (7 points) Recall that a PCFG defines a distribution $p(t)$ over parse trees t. For any sentence s, if we define $\mathcal{T}(s)$ to be the set of valid parse trees for the sentence s, the probability of the sentence under the PCFG is

$$p(s) = \sum_{t \in \mathcal{T}(s)} p(t)$$

Now consider the following PCFG (which is the same as the PCFG in the previous question):

S → NP VP	1.0
VP → V NP	1.0
NP → John	0.6
NP → Mary	0.4
V → saw	1.0

In the space below, write down the parameters of a **trigram language model** that gives the same distribution over sentences as the PCFG shown above.

(Note that each sentence in the trigram language model will be terminated by a STOP symbol, which does not appear at the end of sentences generated by the PCFG.)

(Page intentionally left blank.)

Part #2 8 points

For the following two questions, write TRUE or FALSE below the question. **PLEASE GIVE JUSTIFICATION FOR YOUR ANSWERS: AT MOST 50% CREDIT WILL BE GIVEN FOR ANSWERS WITH NO JUSTIFICATION.**

For all questions in this section we assume as usual that a language model consists of a vocabulary $\mathcal{V}$, and a function $p(x_1 \ldots x_n)$ such that for all sentences $x_1 \ldots x_n \in \mathcal{V}^\dagger$, $p(x_1 \ldots x_n) \geq 0$, and in addition $\sum_{x_1 \ldots x_n \in \mathcal{V}^\dagger} p(x_1 \ldots x_n) = 1$. Here $\mathcal{V}^\dagger$ is the set of all sequences $x_1 \ldots x_n$ such that $n \geq 1$, $x_i \in \mathcal{V}$ for $i = 1 \ldots (n-1)$, and $x_n = \text{STOP}$.

We assume that we have a bigram language model, with

$$p(x_1 \ldots x_n) = \prod_{i=1}^{n} q(x_i | x_{i-1})$$

The parameters $q(x_i|x_{i-1})$ are estimated from a training corpus using linear interpolation. Assume that for any bigram (u, v), $c(u, v)$ is the number of times the bigram (u, v) is seen in the training corpus. In addition assume that for any unigram u,

$$c(u) = \sum_{v \in \mathcal{V} \cup \{\text{STOP}\}} c(u, v)$$

Hence $c(u)$ is the number of times the unigram u is seen as the first word in a bigram.

We assume that for any $u \in \mathcal{V}$, $c(u) > 0$. So every word is seen at least once in the training corpus.

The interpolated estimate is then defined as follows:

$$q(v|u) = \lambda_1(u) \times p_{ML}(v|u) + (1 - \lambda_1(u)) \times p_{ML}(v)$$

where $p_{ML}(v|u)$ and $p_{ML}(v)$ are the bigram and unigram maximum-likelihood estimates, and

$$\lambda_1(u) = \frac{c(u)}{1 + c(u)}$$

We assume throughout this question that all words seen in any test corpus are in the vocabulary $\mathcal{V}$.

Question 4 (4 points) True or False? Under the above definition for $q(v|u)$, for any $u \in \mathcal{V}$, we have

$$\sum_{v \in \mathcal{V} \cup \{\text{STOP}\}} q(v|u) = 1$$

Question 5 (4 points) True or False? For any test corpus, under the above definition for $q(v|u)$, the perplexity under the language model will be less than ∞.

Part #3 18 points

For the following two questions, write TRUE or FALSE below the question. **PLEASE GIVE JUSTIFICATION FOR YOUR ANSWERS: AT MOST 50% CREDIT WILL BE GIVEN FOR ANSWERS WITH NO JUSTIFICATION.**

For all questions in this section we assume as usual that a language model consists of a vocabulary $\mathcal{V}$, and a function $p(x_1 \ldots x_n)$ such that for all sentences $x_1 \ldots x_n \in \mathcal{V}^\dagger$, $p(x_1 \ldots x_n) \geq 0$, and in addition $\sum_{x_1 \ldots x_n \in \mathcal{V}^\dagger} p(x_1 \ldots x_n) = 1$. Here $\mathcal{V}^\dagger$ is the set of all sequences $x_1 \ldots x_n$ such that $n \geq 1$, $x_i \in \mathcal{V}$ for $i = 1 \ldots (n-1)$, and $x_n = \text{STOP}$.

We assume that we have a bigram log-linear language model, with

$$p(x_1 \ldots x_n) = \prod_{i=1}^{n} p(x_i | x_{i-1}; \theta)$$

where the bigram probabilities $p(x_i | x_{i-1}; \theta)$ are defined using a log-linear model. Specifically, the model makes use of a feature vector definition $f(x, y)$, that maps each bigram (x, y) to a feature vector $f(x, y) \in \mathbb{R}^d$, and a parameter vector $\theta \in \mathbb{R}^d$, with

$$p(y|x;\theta) = \frac{\exp(\theta \cdot f(x,y))}{\sum_{y' \in \mathcal{V} \cup \{\text{STOP}\}} \exp(\theta \cdot f(x,y'))}$$

Question 6 (4 points) Given a training corpus consisting of bigrams $(x^{(j)}, y^{(j)})$ for $j = 1 \ldots n$, the parameters are chosen to be

$$\theta^* = \arg\max L(\theta)$$

where

$$L(\theta) = \sum_{j=1}^{n} \log p(y^{(j)}|x^{(j)}; \theta) - \frac{\lambda}{2} \sum_{k=1}^{d} (\theta_k)^2$$

Here $\lambda > 0$ is a positive constant.

True or false? For any test corpus such that every word in the test corpus is in the set $\mathcal{V}$, the perplexity under the parameters θ^* is less than ∞.

Question 7 (4 points) True or false? For any test corpus such that every word in the test corpus is in the set $\mathcal{V}$, there are parameters θ such that the perplexity on the test corpus is $N+1$ where $N = |\mathcal{V}|$.

Question 8 (10 points) If we again define $N = |\mathcal{V}|$, show that it is possible to define a log-linear language model with a single feature (i.e., $d = 1$) such that

$$p(y|x;\theta) = 0.8 \quad \text{if } x = y$$

and

$$p(y|x;\theta) = \frac{0.2}{N} \quad \text{if } x \neq y$$

You should write down your definition for the single feature $f_1(x, y)$, and show the value for the parameter θ_1 that gives the above distribution.

(Page intentionally left blank.)

Part #4

10 points

Question 9 (10 points) In the box below, complete a dynamic programming algorithm that takes as input an integer n, and a probabilistic context-free grammar G, and returns the maximum probability under the PCFG for any tree that has exactly n words.

Hint: the algorithm fills in values for $\pi(i, X)$ for all $i \in \{1 \ldots n\}$, and for all non-terminals X. The value for $\pi(i, X)$ should be the maximum probability for any parse tree with X at the root with exactly i words.

Input: an integer n, a PCFG $G = (N, \Sigma, S, R, q)$ in Chomsky normal form where N is a set of non-terminals, Σ is the set of words, S is the start symbol, R is the set of rules in the grammar, and q is the set of rule parameters.

Initialization:
For all $X \in N$,

$$\pi(1, X) \quad =$$

Algorithm:

- For $i = 2 \ldots n$
  - For all $X \in N$, calculate

$$\pi(i, X) =$$

Output: Return $\pi(n, S)$

Part #5 10 points

Consider a bigram log-linear tagger, where the conditional probability of a tag sequence $y_1 \ldots y_n$ given an input sentence $x_1 \ldots x_n$ is

$$p(y_1 \ldots y_n | x_1 \ldots x_n) = \prod_{i=1}^{n} p(y_i | x_1 \ldots x_n, i, y_{i-1}; \theta)$$

where

$$p(y_i | x_1 \ldots x_n, i, y_{i-1}; \theta)$$

is a log-linear model with parameters θ.

The bigram log-linear tagger defines a function from sentences $x_1 \ldots x_n$ to tag sequences $y_1 \ldots y_n = h(x_1 \ldots x_n)$ as follows:

$$h(x_1 \ldots x_n) = \arg \max_{y_1 \cdots y_n} p(y_1 \ldots y_n | x_1 \ldots x_n)$$

Question 10 (10 points) Assume that we have a bigram log-linear tagger with vocabulary $\mathcal{V} = \{a, b\}$ and a set of possible tags $\mathcal{K} = \{A, B\}$. We would like to build a tagger such that

$$\begin{aligned}
h(a) &= \text{A} \\
h(aa) &= \text{A A} \\
h(aaa) &= \text{A A A} \\
&\ldots \\
h(b) &= \text{B} \\
h(bb) &= \text{B B} \\
h(bbb) &= \text{B B B} \\
&\ldots
\end{aligned}$$

In other words if the input sentence consists of one or more a's, the output of the tagger should be a sequence of all A's. If the input sentence consists of one or more b's, the output of the tagger should be all B's. For sentences that contain both symbols a and b you do not need to worry about the behaviour of the tagger.

In the space below, write down the features and parameters of a bigram log-linear tagger such that it implements the function $h(\ldots)$ described above.

(Page intentionally left blank.)

Part #6 10 points

Assume that we wish to build a log-linear parsing model, using the history-based (Ratnaparkhi) method presented in lectures, which makes use of actions `START(X)` for every non-terminal `X`, `JOIN(X)` for every non-terminal `X`, and `CHECK=YES`/`CHECK=NO`.

Given a log-linear history-based model that defines a distribution

$$p(t|x_1 \dots x_n)$$

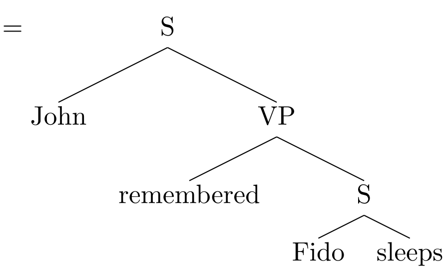
over trees t conditioned on the input sentence $x_1 \dots x_n$, the model defines a function from sentences $x_1 \dots x_n$ to trees $t = h(x_1 \dots x_n)$ as follows:

$$h(x_1 \dots x_n) = \arg\max_t p(t|x_1 \dots x_n)$$

where the $\arg\max$ is taken over all possible parse trees for $x_1 \dots x_n$.

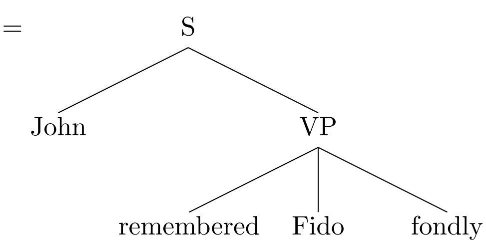
Now assume that we'd like to build a parser such that

$h($`John remembered Fido sleeps`$) =$



and

$h($`John remembered Fido fondly`$) =$



(Continued over the page.)

Question 11 (10 points) Now consider the features of a log-linear model that correctly implements the function $h(x_1 \dots x_n)$. Each feature f_k takes as input a sequence of sub-trees $t_1 \dots t_m$, a position $j \in \{1 \dots m\}$ indicating which sub-tree is the left-most without a `START` or `JOIN` action at its root, and a candidate action a.

For example, for the input *John remembered Fido fondly*, after the sequence of actions

$$\texttt{START(S)}$$
$$\texttt{CHECK} = \texttt{NO}$$
$$\texttt{START(VP)}$$
$$\texttt{CHECK} = \texttt{NO}$$

we have $m = 4$, $j = 3$, and

$$t_1 \dots t_4 = \begin{array}{c}\text{START(S)}\\ |\\ \text{John}\end{array} \quad \begin{array}{c}\text{START(VP)}\\ |\\ \text{remembered}\end{array} \quad \text{Fido} \quad \text{fondly}$$

Complete features 5 and 6 below so that the model can correctly learn the function $h(\dots)$. Assume that we define $t_j = *$ for $j < 1$ or $j > m$, where $*$ is a special symbol. Assume that root(t_j) returns the root symbol for subtree t_j. Assume that additional features may be required in the model for the CHECK actions, and for building higher level structures in the tree.

$$\begin{aligned} f_1(t_1 \dots t_m, j, a) &= 1 \text{ if } root(t_j) = \texttt{John} \text{ and } a = \texttt{START(S)} \\ &= 0 \text{ otherwise} \end{aligned}$$

$$\begin{aligned} f_2(t_1 \dots t_m, j, a) &= 1 \text{ if } root(t_j) = \texttt{remembered} \text{ and } a = \texttt{START(VP)} \\ &= 0 \text{ otherwise} \end{aligned}$$

$$\begin{aligned} f_3(t_1 \dots t_m, j, a) &= 1 \text{ if } root(t_j) = \texttt{sleeps} \text{ and } a = \texttt{JOIN(S)} \\ &= 0 \text{ otherwise} \end{aligned}$$

$$\begin{aligned} f_4(t_1 \dots t_m, j, a) &= 1 \text{ if } root(t_j) = \texttt{fondly} \text{ and } a = \texttt{JOIN(VP)} \\ &= 0 \text{ otherwise} \end{aligned}$$

$$\begin{aligned} f_5(t_1 \dots t_m, j, a) &= 1 \text{ if } root(t_j) = \texttt{Fido} \text{ and } root(t_{j+1}) = \underbrace{\qquad}_{\text{COMPLETE HERE}} \text{ and } a = \underbrace{\qquad}_{\text{COMPLETE HERE}} \\ &= 0 \text{ otherwise} \end{aligned}$$

$$\begin{aligned} f_6(t_1 \dots t_m, j, a) &= 1 \text{ if } root(t_j) = \texttt{Fido} \text{ and } root(t_{j+1}) = \underbrace{\qquad}_{\text{COMPLETE HERE}} \text{ and } a = \underbrace{\qquad}_{\text{COMPLETE HERE}} \\ &= 0 \text{ otherwise} \end{aligned}$$

(Page deliberately left blank.)