Speech and Natural Language Processing **CS60057**
Autumn Semester, 2015 **Mid-Sem**
Maximum Marks: 50 **Time Limit: 2 Hours**

This exam contains 3 pages (including this cover page) and 6 problems.

You may *not* use your books and notes for this exam. Be *precise* in your answers. All the *sub-parts* of a problem should be answered at *one place* only. On multiple attempts, *cross* any attempt that you do not want to be graded for.

There are no clarifications. In case of doubt, you can take a valid assumption, state that properly and continue.

1. (8 points) Suppose that you are given the following sentences
   - Chinese Beijing Chinese
   - Chinese Chinese Shanghai
   - Chinese Macao
   - Tokyo Japan Chinese

   (a) Learn a Bi-gram language model using this data with add-1 smoothing. (5 marks)

   (b) Using the language model learnt in part (a) above, estimate the probability for the sentence, "Chinese Chinese Chinese Tokyo Japan". (3 marks)

2. (6 points) Assume that we have a vocabulary V, i.e., a set of possible words. We would like to estimate a unigram distribution $P(w)$ over $w \in V$. We observe n sample points $w_1, w_2, \ldots, w_n$. Note that this sample may not include all members of V.

   For a word $w \in V$, let $C(w)$ be the number of times it is observed in the training corpus. We now use Good-Turing estimate of its count and define its probability as $P(w) = GT(C(w))/n$. [Let N_r denote the number of members of V which are seen r times in the training corpus.]

   (a) Prove that under this definition, $\Sigma_{w \in V'} P(w) \leq 1$, where V' is the subset of V seen in the training corpus. (3 marks)

   (b) If the "missing" probability mass $1 - \Sigma_{w \in V'} P(w)$ is divided evenly amongst the words not seen in the corpus, find $P(w)$ for any word not in the corpus. (3 marks)

3. (9 points) As per the HMM model for POS tagging, the probability that a tag sequence $t_1, \ldots t_n$ is assigned to the word sequence $w_1, \ldots, w_n$ is given by:

$$P(t_1, \ldots, t_n | w_1, \ldots, w_n) = P(t_1) \prod_{i=1}^{n} P(w_i|t_i) \prod_{i=2}^{n} P(t_i|t_{i-1})$$

   This model corresponds to a bigram tagger.

   (a) Write down the expression for $P(t_1, \ldots, t_n | w_1, \ldots, w_n)$ for a **trigram tagger**. (2 marks)

   (b) Using Maximum Likelihood smoothing, show how each term in the trigram tagging model will be estimated from a training corpus. (2 marks)

(c) Explain how would you modify the Viterbi algorithm used in case of a bigram tagger to handle this case. (5 marks)

4. (8 points) Suppose you want to use a MaxEnt tagger to tag the sentence, "the light book". We know that the top 2 POS tags for the words *the*, *light* and *book* are $\{Det, Noun\}$, $\{Verb, Adj\}$ and $\{Verb, Noun\}$, respectively.

   Assume that the MaxEnt model uses the following history h_i (context) for a word w_i:

   $$h_i = \{w_i, w_{i-1}, w_{i+1}, t_{i-1}\}$$

   where w_{i-1} and w_{i+1} correspond to the previous and next words and t_{i-1} corresponds to the tag of the previous word. Accordingly, the following features are being used by the MaxEnt model:

   - f_1: $t_{i-1} = Det$ and $t_i = Adj$
   - f_2: $t_{i-1} = Noun$ and $t_i = Verb$
   - f_3: $t_{i-1} = Adj$ and $t_i = Noun$
   - f_4: $w_{i-1} = the$ and $t_i = Adj$
   - f_5: $w_{i-1} = the \& w_{i+1} = book$ and $t_i = Adj$
   - f_6: $w_{i-1} = light$ and $t_i = Noun$

   Assume that each feature has a uniform weight of 1.0.

   Use Beam search algorithm with a beam-size of 2 to identify the highest probability tag sequence for the sentence. What is the probability of this sequence?

5. (9 points) We define a PCFG where the non-terminal symbols are $\{S, A, B\}$, the terminal symbols are $\{a, b\}$, and the start symbol is S. The PCFG has the following rules:

Rule	Probability
S → A B	0.2
S → A A	0.2
S → A S	0.2
S → B S	0.4
A → a	0.6
A → b	0.4
B → a	0.7
B → b	0.3

   Use CKY algorithm for PCFG to find the most probable parse tree for the string 'aabbb'.

6. (10 points) Consider two hypothetical word embedding functions, $f(w)$ and $g(w)$, both of which map a word to R^4, but not necessarily the same abstract space. Given below are some words and their embeddings as obtained by f and g, both of which were learnt from English tweets.

Word *w*	*f*(*w*)				*g*(*w*)			
c	4	2	4	0	0	1	10	3
lyk	2	1	1.5	1	10	5	1	7
like	2	1	2	1	7	4	0	6
luk	1	1	1	1	1	0	15	2
luck	1	2	1	1.5	0	7	12	0
lake	2	3	1	3	3	1	4	2
sea	5	1	4	0	1	2	5	3

(a) Given below are two new words and their corresponding f and g embeddings, but not in order. Match the two columns. (3 marks)

Column A	Column B (embedding vectors)				
f(*look*)	[redacted]	0	1	13	3
g(*look*)	[redacted]	4	10	0	1
f(*chance*)	[redacted]	1	1.5	2	1
g(*chance*)	[redacted]	0	5	10	1

(b) From the following list of datasets and features, can you guess what sort of training data and features were used to learn the embeddings f and g? Justify your choice. More than one options can be correct for f as well as g. (4 marks)

Datasets	Features
1. English tweets corpus 2. English newspaper corpus 3. English lexicon 4. English words and their non-standard spellings found in tweets 5. English words and their non-standard spellings found in SMS. 6. Parts-of-speech tagged tweet corpus 7. English tweets and their translations.	A. Character n-grams B. Word n-grams C. The previous and next k words around a word D. Presence of a word in the lexicon E. Parts-of-speech of a word F. Whether the first character is capitalized G. Length of the tweet

(c) Suppose $d(x, y)$ denotes the Euclidian distance between two vectors x and y. If the two words w and w' are orthographic variations of each other, then which of the following statements are most likely to be true? Justify your answer. (1+2 marks)

(i) $d(f(w), f(w'))/d(g(w), g(w'))$ is small

(ii) $d(f(w), f(w'))^{*}d(g(w), g(w'))$ is small

(iii) $d(f(w), f(w')) + d(g(w), g(w'))$ is small

(iv) $d(f(w), f(w')) - d(g(w), g(w'))$ is small