CS6320, Spring 2018
Dr. Mithun Balakrishna
Homework 2
Due Wednesday, February 21st, 2018 11:59pm

A. Submission Instructions:

- Submit your solutions via eLearning.
- Please submit a single zip file with the following files:
  - For programming questions:
    - Source code file(s) in C/C++, Java, or Python. For using any other programming language, please get prior approval from the TA.
    - A ReadMe file with instructions on how to compile/run the code.
  - For all other questions, a PDF/Doc/PS/Image file with the solutions.
- Late Submission Penalty:
  - up to 2 hours late — 10% deduction
  - 2 - 4 hours late — 20% deduction
  - 4 - 12 hours late — 35% deduction
  - 12 - 24 hours late — 50% deduction
  - 24 - 48 hours late — 75% deduction
  - more than 48 hours late — 100% deduction (zero credit)

B. Problems:

1. Regular Expression (20 points)

Write a single regular expression for identifying social security numbers in text. The social security numbers consists of:

- 9 digits
- must be preceded by one or more spaces or beginning of line
- must be followed by one or more spaces or end of line

In addition there are certain restrictions:

- first three digits cannot be all zeros
- last four digits cannot be all zeros
- nine digits can all be next to each other

  or

  there can be a hyphen between:
  - third and fourth digit, and
  - fifth and sixth digit

The following are well formed social security numbers: 123456789, 001-00-7089.

The following are ill-formed social security numbers: 000-23-4567, 123-45-0000, 001-007089, 00100-7089.

There is no valid social security number on the following line:
12345678910 is a big number, 345-678-910 is a lotto number and 3333333334 is a 10 digit number.

2. Bigram Probabilities (40 points):

An automatic speech recognition system has provided a written sentence as the possible interpretation to a speech input.

Compute the probability of a written sentence using the bigram language model trained on *HW2_F17_NLP6320-NLPCorpusTreebank2Parts-CorpusA.txt* (provided as Addendum to this homework on eLearning).

Note: Please use whitespace (i.e. space, tab, and newline) to tokenize the corpus into words/tokens that are required for the bigram model. Do NOT perform any type of word/token normalization (i.e. stem, lemmatize, lowercase, etc.). Creation and matching of bigrams should be exact and case-sensitive. Do NOT split the corpus into sentences. Please consider the entire corpus as a single string for tokenization and computation of bigrams.

Compute the sentence probability under the three following scenarios:

i. Use the bigram model without smoothing.
ii. Use the bigram model with add-one smoothing
iii. Use the bigram model with Good-Turing discounting.

Your computer program should do the following:

1. Compute the bigram counts on the given corpus (*HW2_F17_NLP6320-NLPCorpusTreebank2Parts-CorpusA.txt*).
2. For a given input written sentence:
   a. For each of the three scenarios, construct a table with the bigram counts for the sentence.
   b. For each of the three scenarios, construct a table with the bigram probabilities for the sentence.
   c. For each of the three scenarios, compute the total probability for the sentence.

3. Transformation Based POS Tagging (40 points)

For this question, you have been given a POS-tagged training file, *HW2_F17_NLP6320_POSTaggedTrainingSet.txt* (provided as Addendum to this homework on eLearning), that has been tagged with POS tags from the Penn Treebank POS tagset (Figure 1).

Tag	Description	Example	Tag	Description	Example
CC	coordin. conjunction	*and, but, or*	SYM	symbol	+,%, &
CD	cardinal number	*one, two, three*	TO	“to”	*to*
DT	determiner	*a, the*	UH	interjection	*ah, oops*
EX	existential ‘there’	*there*	VB	verb, base form	*eat*
FW	foreign word	*mea culpa*	VBD	verb, past tense	*ate*
IN	preposition/sub-conj	*of, in, by*	VBG	verb, gerund	*eating*
JJ	adjective	*yellow*	VBN	verb, past participle	*eaten*
JJR	adj., comparative	*bigger*	VBP	verb, non-3sg pres	*eat*
JJS	adj., superlative	*wildest*	VBZ	verb, 3sg pres	*eats*
LS	list item marker	*1, 2, One*	WDT	wh-determiner	*which, that*
MD	modal	*can, should*	WP	wh-pronoun	*what, who*
NN	noun, sing. or mass	*llama*	WP$	possessive wh-	*whose*
NNS	noun, plural	*llamas*	WRB	wh-adverb	*how, where*
NNP	proper noun, singular	*IBM*	$	dollar sign	$
NNPS	proper noun, plural	*Carolinas*	#	pound sign	#
PDT	predeterminer	*all, both*	“	left quote	‘ or “
POS	possessive ending	*’s*	”	right quote	’ or ”
PRP	personal pronoun	*I, you, he*	(	left parenthesis	[, (, {, <
PRP$	possessive pronoun	*your, one’s*	)	right parenthesis	],), }, >
RB	adverb	*quickly, never*	,	comma	,
RBR	adverb, comparative	*faster*	.	sentence-final punc	. ! ?
RBS	adverb, superlative	*fastest*	:	mid-sentence punc	: ; ... – -
RP	particle	*up, off*			

Figure 1. Penn Treebank POS tagset

Use the POS tagged file to perform:

a. Transformation-based POS Tagging: Implement Brill’s transformation-based POS tagging algorithm using ONLY the previous word’s tag to create transformation rules.

b. Naïve Bayesian Classification (Bigram) based POS Tagging:

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}}\, P(t_1^n|w_1^n) \approx \underset{t_1^n}{\operatorname{argmax}} \prod_{i=1}^{n} P(w_i|t_i)P(t_i|t_{i-1})$$

c. Apply model (a) and (b) on the sentence below, and show the difference in error rates.

Sentence: *The president wants to control the board 's control*

Manual POS Tagged Sentence: *The*_DT *president*_NN *wants*_VBZ *to*_TO *control*_VB *the*_DT *board*_NN *'s*_POS *control*_NN