Final Exam
Natural Language Processing

May 30, 2016

Remember to fill in your name on all pages
Good luck

Problem 1 (1.5 points). Compute the Good-Turing probability of the sentence *Who is April?* based on the following corpus:

My name is April. April is my name. My April was born in April.

Make sure you include punctuation signs in your calculations and ignore capitalization (that is, *My* and *my* represent the same word).

Problem 2 (2 points). Assume you are given the task to pick up the most probable correction for the word *acress* among two possible candidates: *actress* and *across* by using a Noisy channel model. For this, you will have to fill in the following table (x = *acress* and w is either *actress* or *across*).

Candidate Correction	Correct Letter	Error Letter	Type	$P(x\|w)$	$P(w)$	$P(x\|w) * P(w)$
actress across						

The language model used is a unigram model based on the following corpus (ignore punctuation signs in your calculations):

My favorite actress walked across the field to approach me. It never crossed my mind that an actress might be interested in meeting me.

The channel model is build based on the following list of typical errors.

affectionate: affecient	*director: directer*	*predictable: predicable*
affects: affets	*factor: facter*	*previous: previos*
collection: coletion	*object: objet*	*reconnect: reconnet*
capitol: capital	*patch: pacth*	*restriction: restricion*

In the *Type* column you need to indicate whether x can be obtained from w by deleting a letter l (in which case, Correct Letter is l and Error Letter is -), by inserting a letter l (in which case, Correct Letter is - and Error Letter is l), by substituting the letter l_1 with letter l_2 (in which case, Correct Letter is l_1 and Error Letter is l_2) or by transposing the group of letters l_1l_2 (in which case, Correct Letter is l_1l_2 and Error Letter is l_2l_1).

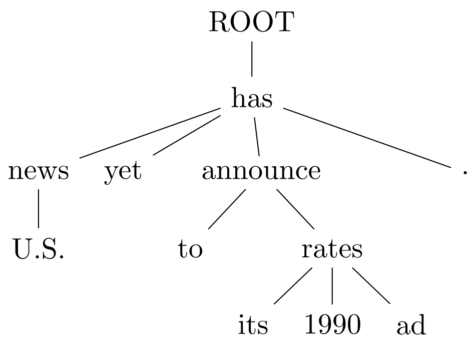
Problem 3 (1 point). What is the mean average precision (MAP) for the following sequence of retrieved documents, where R denotes a relevant document and N denotes an irrelevant document? (Assume there are 25 relevant documents in the collection)

N R N R N R N N R N R N R N R

Problem 4 (2 points). Write a **minimal** definite clause grammar (DCG) in prolog with the following vocabulary: *la, las, el, los, un, una, unos, unas, perra, perras, perro, perros, hueso, huesos, estudiante, bonita, bonito, bonitas, bonitos, ladra, ladran, muerde, muerden*.

The grammar should accept all grammatical sentences, even if they are meaningless (for example, it should accept "la estudiante bonita muerde una perra"), but it should not accept ungrammatical entries such as "los perras muerden un estudiante" o "un perro ladra un hueso". Note that you will need to distinguish between transitive verbs (such as "muerde") and intransitive verbs (such as "ladra").

Problem 5 (2.5 points). Given the English sentence *U.S. news has yet to announce its 1990 ad rates.*, simulate the run of the Basic Dependency Parser in order to get its dependency tree (which you can find below).



Problem 6 (1 point). Let's say we've calculated ppmi(*Stanford*, *University*), that is the positive pointwise mutual information for the word *Stanford* in the context of *University*, and found that to be 2.3219. The particular context we are examining is one in which *University* was the next word following *Stanford*, though for this problem, you don't need to be concerned with how the specific context is defined. Your professor now wants you to find how many of the sentences you examined contained the word *Stanford*. Rather than running through the entire corpus and searching for the word *Stanford*, you instead attempt to calculate this count using numbers you noted from before.

You remember looking at a corpus with a total number of 100,000 tokens. Also, you observed that there was a 50% chance that you saw *Stanford* right before the word *University* in the sentence when a sentence contained *University*. For the sake of simplicity, also assume that each sentence contained at most one instance of the word *Stanford* or *University*. How many times did the word *Stanford* appear in your corpus? Assume that the ppmi was calculated using a log of base 2 and round your answer to the nearest integer.

DEFINITION 1. Let w be a word that appears c times in the corpus, N_c = the count of things we've seen c times, and N = the total number of tokens in the corpus. Then,

$$c^*(w) = \begin{cases} \frac{(c+1)N_{c+1}}{N_c}, & \text{if } c > 0 \\ N_1, & \text{if } c = 0 \end{cases}$$

$$P^*_{GT}(w) = \frac{c^*(w)}{N}$$

The Good-Turing probability of a sentence $w_1 \dots w_n$ is obtained by multiplying individual probabilities:

$$P^*_{GT}(w_1 \dots w_n) = \prod_{i=1}^{n} P^*_{GT}(w_i).$$

DEFINITION 2. Computing error probability:

$del[x, y]$: count(xy typed as x) $sub[x, y]$: count(x typed as y)
$ins[x, y]$: count(x typed as xy) $trans[x, y]$: count(xy typed as yx)

$$P(x \mid w) = \begin{cases} \frac{del[w_{i-1},w_i]}{count(w_{i-1}w_i)}, & \text{if } x = w_1 \dots w_{i-1}w_{i+1} \dots w_n \text{ and } w = w_1 \dots w_{i-1}w_i w_{i+1} \dots w_n \\ \frac{ins[w_{i-1},x_i]}{count(w_{i-1})}, & \text{if } x = w_1 \dots w_{i-1}x_i w_i \dots w_n \text{ and } w = w_1 \dots w_{i-1}w_i \dots w_n \\ \frac{sub[w_i,x_i]}{count(w_i)}, & \text{if } x = w_1 \dots w_{i-1}x_i w_{i+1} \dots w_n \text{ and } w = w_1 \dots w_{i-1}w_i w_{i+1} \dots w_n \\ \frac{trans[w_i,w_{i+1}]}{count(w_i w_{i+1})}, & \text{if } x = w_1 \dots w_{i+1}w_i \dots w_n \text{ and } w = w_1 \dots w_i w_{i+1} \dots w_n \end{cases}$$

In the unigram model, $P(w) = \frac{count(w)}{N}$, where N is the total number of tokens.

DEFINITION 3. The Mean Average Precision (MAP) is the average of the precision value obtained for the top k documents, each time a relevant document is retrieved.

Precision = TP/(TP+FP)
TP = true positives (the number of gold answers that were correctly guessed)
FP = false positives (the number of guessed answers that were incorrect)
FN = false negative (the number of gold answers that were not correctly guessed)

DEFINITION 4. Writing a DCG in prolog (using extra arguments) can be done using the following syntax:

```
s --> np(subject), vp.          det --> [the].
np(_) --> det, n.               n --> [woman].
np(X) --> pro(X).               v --> [shoots].
vp --> v, np(object).           pro(subject) --> [he].
vp --> v.                       pro(object) --> [him].
```

DEFINITION 5. The Basic Dependency Parser:

Start: σ = [ROOT], $\beta = w_1, \dots, w_n$, $A = \emptyset$

1. Shift	$\sigma, w_i \mid \beta, A$	$\sigma \mid w_i, \beta, A$
2. Left-Arc$_r$	$\sigma \mid w_i, w_j \mid \beta, A$	$\sigma, w_j \mid \beta, A \cup \{r(w_j, w_i)\}$
3. Right-Arc$_r$	$\sigma \mid w_i, w_j \mid \beta, A$	$\sigma, w_i \mid \beta, A \cup \{r(w_i, w_j)\}$

Finish: $\beta = \emptyset$

DEFINITION 6. Positive Pointwise Mutual Information:

$$ppmi(w_1, w_2) = \begin{cases} \log_2 \frac{P(w_1,w_2)}{P(w_1)P(w_2)}, & \text{if } P(w_1, w_2) \geq P(w_1)P(w_2) \\ 0 & \text{otherwise} \end{cases}$$

$P(w_1, w_2) = \frac{count(w_1,w_2)}{N}$ and $P(w) = \frac{count(w)}{N}$, where N is the number of tokens.