Final Exam
Natural Language Processing

September 8, 2016

Remember to fill in your name on all pages
Good luck

Problem 1 (2.5 points). We are given the following corpus:

- <s> My name is April </s>
- <s> April is my name </s>
- <s> My April was born in April </s>
- <s> Who is born in April </s>

Using interpolated Kneser-Ney smoothing, what is P_{KN}(<s> Who is April </s>) if we use a discount factor of $d = 0.5$?

Problem 2 (2 points). Suppose we have the following short movie reviews, each labeled with a genre, either **comedy** or **action**.
1. *fun, couple, love, love* (**comedy**)
2. *fast, furious, shoot* (**action**)
3. *couple, fly, fast, fun, fun* (**comedy**)
4. *furious, shoot, shoot, fun* (**action**)
5. *fly, fast, shoot, love* (**action**)

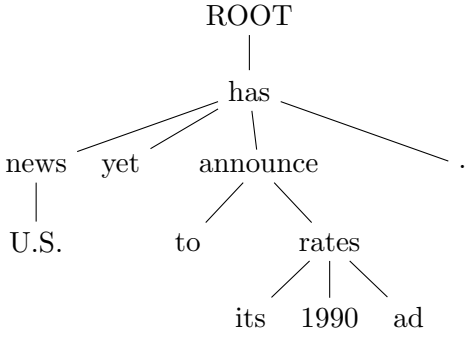
Using a simple Naïve Bayes approach with Laplace smoothing, how would we classify the following review: *fast, furious, love, comedy*?

Problem 3 (1 point). Compute the Minimal Edit Distince between "marble" and "blender" using dynamic programming.

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 "Arc-eager" Dependency Parser in order to get its dependency tree (which you can find below).



DEFINITION 1. The Kneser-Ney probability of a sentence $w_0 w_1 \dots w_n$ is given by:

$P_{KN}(w_0 w_1 \dots w_n) = \sum_{i=1}^{n} \log_{10} P_{KN}(w_i \mid w_{i-1})$

$P_{KN}(w_i \mid w_{i-1}) = \frac{\max(c(w_{i-1}, w_i) - d, 0)}{c(w_{i-1})} + \lambda(w_{i-1}) P_{CONT}(w_i)$

$\lambda(w_{i-1}) = \frac{d}{c(w_{i-1})} \mid \{w \mid c(w_{i-1}, w) > 0\} \mid$

$P_{CONT}(w) = \frac{|\{w' | c(w', w) > 0\}|}{|\{(w', w'') | c(w', w'') > 0\}|} = \frac{|\{w' | c(w', w) > 0\}|}{\sum_{w''} |\{w' | c(w', w'') > 0\}|}$

DEFINITION 2. Let S be a sentence, C a class of possible labels and V the vocabulary. Then,

$$c_{NB}(S) = argmax_{c \in C} P(c) \prod_{w \in S} P(w \mid c),$$

where the prior probability of the class is $P(c) = doccount(c)/N_{doc}$, and the probability of a word w given the class c, computed using Laplace (add-1) smoothing with unknown words, is:

$$P(w \mid c) = \frac{count(w, c) + 1}{\sum_{v \in V} count(v, c) + |V| + 1}$$

DEFINITION 3. A dynamic programming algorithm that computes Minimal Edit Distance: for two words w and w' of length n and m, respectively, we compute $D(i, j)$ for small i, j, and then larger $D(i, j)$ based on previously computed smaller values, where $D(i, j) =$ min edit distance between the i-length prefix of w and the j-length prefix of w'.

- Initialization
  $D(i, 0) = i, \forall i \in 0, \dots, n$
  $D(0, j) = j, \forall j \in 0, \dots, m$
- Recurrence Relation:

  For each $i = 1, \dots, n$

  For each $j = 1, \dots, m$

  $D(i, j) = \min(D(i-1, j) + 1, D(i, j-1) + 1, D(i-1, j-1) + d)$,

  where $d = \begin{cases} 2, & \text{if } w_i \neq w'_j \\ 0, & \text{if } w_i = w'_j \end{cases}$
- Termination
  Output $D(n, m)$

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 "Arc-eager" Dependency Parser:

Start: $\sigma = [\text{ROOT}]$, $\beta = w_1, \dots, w_n$, $A = \emptyset$

1. Shift	$\sigma, w_i \mid \beta, A$	$\sigma \mid w_i, \beta, A$
Precondition:	$r'(w_k, w_i) \notin A$,	$w_i \neq$ ROOT
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$_r$	$\sigma \mid w_i, w_j \mid \beta, A$	$\sigma \mid w_i \mid w_j, \beta, A \cup \{r(w_i, w_j)\}$
Precondition:	$r'(w_k, w_i) \in A$	
4. Reduce	$\sigma \mid w_i, \beta, A$	σ, β, A

Finish: $\beta = \emptyset$