Final Exam
Natural Language Processing

September 1, 2017

Remember to fill in your name on all pages
Good luck

Problem 1 (1 point). Compute the Minimal Edit Distance between "monkey" and "cookie" using dynamic programming.

Problem 2 (3 points). In a set of 100,000 documents, we get the following data on a few terms and a few documents:

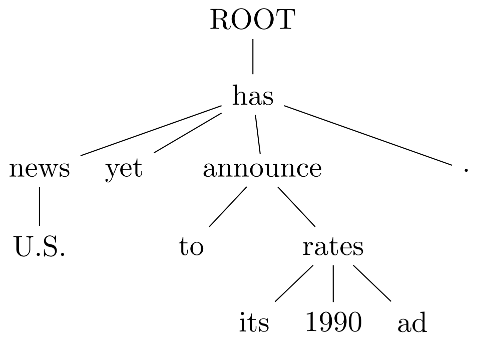
term	document frequency	Doc1	Doc2	Doc3
computer	10,000	33	2	55
data	100	4	42	0
desktop	10	0	27	3
cheap	1,000	15	10	22

Compute the tf-idf value for these terms and documents. What is the cosine similarity between query "cheap desktop computer" and Doc 1, 2 and 3, respectively (use tf-idf weighting - ltn.lnc variation). Which of these three documents would be ranked first by a search engine using the ltn.lnc scheme?

Problem 3 (2 points). Write a **minimal** definite clause grammar (DCG) in prolog with the following vocabulary: *the, cat, cats, dog, dogs, like, likes, milk, sugar*.

The grammar should accept all grammatical sentences, even if they are meaningless (for example, it should accept "the sugar likes milk"), but it should not accept ungrammatical entries such as "cat like milk".

Problem 4 (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).



Problem 5 (1.5 points). Define polysemy/metonymy, synonymy, antonyms and hyponymy/hypernymy. Give an illustrative example for each term.

Definition 1. 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 2. SMART Notation: denotes the combination in use in an engine, with the notation ddd.qqq, using the acronyms from the following table:

Term frequency	Document frequency	Normalization
n (natural): $tf_{t,d}$	n (no): 1	n (none): 1
l (logarithm): $1 + \log_{10}(tf_{t,d})$	t (idf): $\log_{10}(\frac{N}{df_t})$	c (cosine): $\frac{1}{\sqrt{\sum_i x_i^2}}$

Definition 3. 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 4. The "Arc-eager" Dependency Parser:

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

1. Shift	$\sigma, w_i \mid \beta, A$	$\sigma \mid w_i, \beta, A$
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: $\beta = \emptyset$