Practical Exercise 4: Probabilistic Context Free Grammars

In this practical, you will implement the Inside-Outside algorithm seen in class. The goal is to construct 3 programs that answer the following questions:

- (Q1) Given a probabilistic grammar G and an input sentence S, compute the probability that grammar G generates S.
- (Q2) Given a probabilistic grammar G and an input sentence S, compute the parse of S with the highest probability (w.r.t. G).
- (Q3) Given a corpus (sequence of sentences), generate a locally max-likelihood grammar for the corpus.

Note that we are ignoring the need for rescaling numbers (to avoid underflow) which may be important in some applications.

1 The Grammar

We will use the grammar in the book (Manning & Schütze, page 384), slightly modified. We refer to this grammar as $G_{M&S}$.

Nonterminals are S, NP, PP, VP, P, V. The start symbol is S.

Terminals are with, saw, are, eat, astronomers, ears, stars, telescopes.

Rules are:

Pr[S -> NP VP] = 1.0
Pr[PP -> P NP] = 1.0
Pr[VP -> V NP] = 0.7
Pr[VP -> VP PP] = 0.3
Pr[P -> with] = 1.0
Pr[V -> saw] = 0.5
Pr[V -> are] = 0.4
Pr[V -> eat] = 0.1
Pr[NP -> NP PP] = 0.4
Pr[NP -> astronomers] = 0.1
Pr[NP -> ears] = 0.18
Pr[NP -> saw] = 0.04
Pr[NP -> stars] = 0.18
Pr[NP -> telescopes] = 0.1

We provide the data structure for the grammar. You can find the files grammar.cpp and grammar.h in the directory /g/150TP/files/PP4/. These files implement the class grammar in C++. The header file grammar.h explains how to use it.
2 The Corpus

The file /g/150TP/files/PP4/corpus.txt contains input sentences, one in each line. These sentences contain only terminals of the grammar described above.

3 What you should do

Write the following programs:

- q1 <corpus-file> This program takes the grammar $G_{M&S}$ and computes the probability of every sentence in the corpus specified. Use the $\beta_j(p, q)$ table as described in the book to compute this probability. Output a line for each sentence, e.g.:

  Pr[astronomers saw stars with ears] = 0.0015876
  Pr[astronomers are stars] = 0.00504
  Pr[stars are ears with ears with ears with stars] = 3.6771e-05
  ...

- q2 <corpus-file> This program takes the grammar $G_{M&S}$ and computes the probability of the most likely parse of the first sentence in the corpus specified. Use the $\delta_j(p, q)$ table to compute this probability and the $\psi_j(p, q)$ to trace back the parse tree, as described in the book. It should output the probability of the most likely parse, the parse tree, and the $\delta_j(p, q)$ table (output non-zero entries only). A plain text description of the parse tree suffices, e.g.:

  Pr[most likely parse of "astronomers saw stars with ears"] = 0.0009072

Parse is (depth-first traverse of the tree):
S -> NP VP
NP -> astronomers
VP -> V NP
V -> saw
NP -> NP PP
NP -> stars
PP -> P NP
P -> with
NP -> ears

Delta table is:
delta[S][1][3] = 0.0126
delta[S][1][5] = 0.0009072
delta[NP][1][1] = 0.1
delta[NP][2][2] = 0.04
delta[NP][3][3] = 0.18
delta[NP][3][5] = 0.01296
delta[NP][5][5] = 0.18
delta[VP][2][3] = 0.126
delta[VP][2][5] = 0.009072
delta[PP][4][5] = 0.18
delta[V][2][2] = 1
delta[P][4][4] = 1

• q3 <corpus-file> <init-mode> This program should implement the EM algorithm. The initial grammar should be initialised according to the <init-mode> specified. This initialisation mode can be either –random or –MS. If the mode is –random, initialise probabilities randomly. If the mode is –MS, initialise probabilities according to $G_{M&S}$. The program should go through the sentences in <corpus-file> repeatedly, updating the probabilities of the grammar after each pass, until none of the probabilities in the grammar changes more than $\epsilon = 0.001$ or you have gone over the corpus 100 times. Use the $\alpha_j(p, q)$ and $\beta_j(p, q)$ tables to compute the new probabilities of the grammar $P[N^j \rightarrow N^TN^s]$ and $P[N^j \rightarrow w^k]$ for the whole corpus following the formulae in the book (Manning & Schütze, page 400-401). Your program should print the probability of the corpus after every pass through the corpus and the initial and final grammar, e.g.:

Initial grammar is:
Pr[S -> S S] = 0.0220027
Pr[S -> S NP] = 0.0363718
Pr[S -> S VP] = 0.0323305
.
Pr[P -> astronomers] = 0.0353581
Pr[P -> ears] = 0.0335449
Pr[P -> stars] = 0.00861287
Pr[P -> telescopes] = 0.0389846

Probabilities of corpus:
Pr[after pass 0 of 'corpus.txt'] = 9.79314e-61
Pr[after pass 1 of 'corpus.txt'] = 3.68011e-08
Pr[after pass 2 of 'corpus.txt'] = 7.56291e-07
.
Pr[after pass 8 of 'corpus.txt'] = 0.00227437

Final grammar is:
Pr[S -> S NP] = 0.299408
Pr[S -> S VP] = 9.07136e-112
.
Pr[P -> with] = 5.45154e-291
Pr[P -> are] = 0.333227
Pr[P -> telescopes] = 0.66656
4 What to hand in

Please submit your solution electronically to marias@eecs.tufts.edu before Wednesday, 2nd May 2001, 5pm. You should include in your message:

- Output given by programs q1 and q2 when run on corpus /g/150TP/files/PP4/corpus.txt.
- Output given by call to q3 /g/150TP/files/PP4/corpus.txt -MS.
- Outputs given by 2 calls to q3 /g/150TP/files/PP4/corpus.txt -random (each output in a separate file).
- Clear, comprehensive documentation on how your programs work.
- Your code.

All the files handed in should be text files.