DALHOUSIE UNIVERSITY *Inspiring Minds*

Winter 2020 (Jan6-Apr6)
Faculty of Computer Science

CSCI 4152/6509 — Natural Language Processing

Assignment 4

Due: *Friday Apr 17, 2020 by midnight*
Worth: 150 marks $(= 24 + 50 + 20 + 35 + 21)$
Instructor: Vlado Keselj, CS bldg 432, 902.494.2893, vlado@dnlp.ca

---

Assignment Instructions:

The submission process for Assignment 3 is mostly based on the `submit-nlp` command on `bluenose` as discussed in the lab, or in the equivalent way by using the course web site, where you need to follow 'Login' and then the 'File Submission' menu option. Some questions may have some specific different submission instructions, in which case you should follow those.

Important: You must make sure that your course files on bluenose are not readable by other users. For example, if you keep your files in the directory `csci6509` or `csci4152` you can check its permission using the command:

```
    ls -ld csci6509
or  ls -ld csci4152
```

and the output must start with `drwx------`. If it does not, for example if it starts with `drwxr-xr-x` or similar, then the permissions should fixed using the command:

```
    chmod 700 csci6509
or  chmod 700 csci4152
```

1) (24 marks) Complete the Lab 9 as instructed. In particular, you will need to properly:

a) (4 marks) Submit the file '`parse.prolog`' as instructed.

b) (4 marks) Submit the file '`dcg.pl`' as instructed.

c) (4 marks) Submit the file '`dcg-ptree.prolog`' as instructed.

d) (4 marks) Submit the file '`dcg-agr.prolog`' as instructed.

e) (4 marks) Submit the file '`dcg-pcfg.prolog`' as instructed.

f) (4 marks) Submit the file '`dcg-agr2.prolog`' as instructed.

2) (50 marks) Submit your solution as a file named either as `a4q2.txt` if it is a plain-text file, `a4q2.pdf` if it is a PDF file, or `a4q2.jpg` if it is an image file. You can type your solution as a plain text file, prepare in electronic form, or write it nicely on paper and take pictures. If you need to submit more than one picture, then number them as follows: `a4q2-1.jpg`, `a4q2-2.jpg`, etc.

Consider the following five sentences tagged with POS tags:

```
time N flies V like P an D arrow N
time N flies N like V an D arrow N
arrow N flies V like P time N
flies N like V flies N
an D arrow N like V flies N
```

a) (20 marks) Calculate necessary CPTs (Conditional Probability Tables) for the HMM model. *Do not apply any smoothing.*

b) (10 marks) Consider tagging the sentence: "`arrow flies like arrow`"

Draw the factor graph for this problem and mark all messages that need to be computed. It is probably hard to do this in plain text, so you can make it a part of a pdf file, or jpg file with a name as specified at the beginning of the question. You can even draw the graph by hand and submit a photo of it.

c) (20 marks) Calculate all necessary messages and find the optimal values for T_1, T_2, T_3, and T_4. Show values of all messages and calculation needed to find optimal values of the variables.

Of course, you need to use the unsmoothed tables from part a). There are no unseen words in the given sentence, so smoothing or some other way of handling unseen words is not necessary.

You are encouraged to write a program for some of these tasks, but you must submit output of the program with sufficient details as required because you will be marked on the output.

3) (20 marks) Submit your answer either as plain text in one of the options: a file `a4q3.txt`, PDF file named `a4q3.pdf`, and image file named `a4q3.jpg`, or a few image files called `a4q3-1.jpg`, `a4q3-2.jpg`, etc.

Consider the following two sentences:

The author walked in the city with Jane.
and
The author lived in the city with large population.

The have different structure because "with Jane" in the first sentence refers to who the author walked with, while in the second sentence, "with large population" refers to the city. This difference can be clearly shown if we properly construct parse trees of the two sentences.

a) (10 marks) Draw the parse trees of the sentences using the standard tags and rules.
b) (5 marks) Annotate the **second** parse tree with head information and dependencies. (as done in the example in class)
c) (5 marks) Re-write the **second** parse tree using the bracketed notation.

4) (35 marks) Submit your answer using the `submit-nlp` command, and use one of the following options: a plain text file named `a4q4.txt`, or a PDF file named `a4q4.pdf`, or an image file named `a4q4.jpg`, or several image files named `a4q4-1.jpg`, `a4q4-2.jpg`, etc.

a) (15 marks) Extract the context-free grammar induced from the following parse trees. If we express this grammar as a PCFG, what are the probabilities that can be inferred from the trees?

```
(S (WNP (WDT What)
        (NN courses))
   (VP (BE are)
       (VP (VBN offered)
           (PP (IN in)
               (NN fall)))))

(S (WNP Who)
   (VP (VBZ teaches)
       (NP (NN CSCI)
           (NN 1100))))

(S (WNP (WDT (WRB How) (JJ many))
        (NN students))
   (VP (BE are)
       (VB (VBG taking)
           (NP (NN CSCI)
               (NN 1108)))))
```

b) (15 marks) Using CYK algorithm for completion and the grammar obtained in the part a), parse the sentence: *Who are taking CSCI courses*

Your answer must include a fully computed CYK chart.

c) (5 marks) What is the final parse tree obtained from parsing in the part b) and what is the probability of this tree?

5) (21 marks, programming) Write and submit a program written in Perl, Python, C, C++, or Java, named `a4q5.pl`, `a4q5.py`, `a4q5.c`, `a4q5.cc`, or `a4q5.java`, which verifies one or more bracketed parse trees, and check that their grammar is the Chomsky Normal Form grammar.

The program must read the standard input. It expects one or more parse trees in the input and the program should process them. The parse trees are in the format discussed in the class. You should also assume that all non-terminals consist of uppercase letters and terminals consist of uppercase letters, lowercase letters, and digits, and not necessarily all of them. In other words, terminals satisfy the regular expression `/[A-Za-z0-9]+/`. Each parse tree will start with open parenthesis '`(`', and there can be spaces between parentheses, terminals, and non-terminals.

If the trees are all valid and their grammar is in Chomsky Normal Form, the program must print the following line:
`Valid CNF trees.`
(Always finish a line with a new-line character.)

For example, the following input:

```
(S (NP dogs) (VP run)) (S nothing)
```

should produce output:

```
Valid CNF trees.
```

If the trees do not follow specifications, are not proper trees, or not in Chomsky Normal Form, then the program must print the line:
`Not valid CNF trees.`

If there are more than one trees, and even if only one of those trees is not in the CNF form, your program should still report the '`Not valid CNF trees`' message. For example, the following input should produce this message:

```
(S (NP dogs) (VP run) (ADV fast))
```

As a more complex example, the following input should be accepted as 'Valid CNF trees':

```
(S (WNP (WDT What)
        (NN courses))
   (VP (BE are)
       (VP (VBN offered)
           (PP (IN in)
               (NN fall)))))
```

```
(S (WNP Who)
   (VP (VBZ teaches)
       (NP (NN CSCI)
           (NN 1100))))

(S (WNP (WDT (WRB How) (JJ many))
        (NN students))
   (VP (BE are)
       (VB (VBG taking)
           (NP (NN CSCI)
               (NN 1108)))))
```