The University of Texas at Dallas
CS 6320
Natural Language Processing
Spring 2020
Instructor: Dr. Sanda Harabagiu
Grader/ Teaching Assistant: Maxwell Wienzierl

Homework 2: 100 points

Issued March 10, 2021
Due April 7, 2021 before midnight

PROBLEM 1: Sentiment Detection (**30 points**)
The main goal of this problem is to test your ability to perform sentiment detection by executing software publicly available on github on the Google cloud. This time you will prepare your own framework (in the same way it was prepared for you in Homework 1 for Problem 2.B) and train and test the system in your google Colab account.

The github repository you should use is found at:
https://github.com/uzaymacar/comparatively-finetuning-bert

The task is to try performing sentiment detection on IMDB movie reviews. Code and instructions are provided in the github repo. Download the code and data to a Google Colab notebook, acquire a GPU runtime, install necessary python packages, and run the code. Training should take approximately 150 minutes, with each epoch taking approximately 15 minutes. Document the training and test loss and accuracy at each epoch and graph the curves in your report. Discuss challenges you ran across while setting up the environment and comment on the model's train and test performance.

PROBLEM 2: Parsing with your Cocke-Kasami-Younger (CKY) parser (**50 points**)
Generate an automatic CKY parser for the following sentences:
S1: Sales of the company to return to normalcy.
S2: The new products and services contributed to increase revenue.
S3: Dow falls as recession indicator flashed red and economical worries continue through the month.

S4: Figure skater lands historic quadruple jump in senior international competition at the 2019 World Figure Skating Championships on Day 3 but could only clinch a silver medal.

You should generate a grammar for all non-terminals as well as all lexical terminals in the four sentences you need to parse. Convert by hand your grammar to Chomsky Normal Form (CNF) (**10 points**).
Write a program that should do the following:

1. Load the CNF grammar.
2. Read in the example sentences,
3. For each example sentence, output to a file:
    - o the sentence itself
    - o the simple bracketed structure parse(s) based on your implementation of the CKY algorithm, and
    - o the number of parses for that sentence.

Running your code should be performed from:
Hw2_CKYparser.{py|java|etc} <grammar_filename> <sentence_filename> <output_filename>
where:

- o <grammar_filename> is the name of the file holding grammar rules in the in Chomsky Normal Form.
- o <test_sentence_filename> is the name of the file containing four test sentences to parse with your algorithm.
- o <output_filename> is the name of the file where your system will write the parses and over the test sentences.

You will receive **10 points** for each sentence that is automatically parsed by your program.

PROBLEM 3: Dependency Parsing (**20 points**)
1] Considering the same four sentences as in Problem 2, generate the dependency parses of each of these sentences using two different dependency parsing models:

You can use both python spacy dependency parse models:
en_core_web_sm - Statistical dependency parser
en_core_web_trf – Neural transformer dependency parser (roberta-base)

Spacy can be installed as a python packages as pip install spacy, and each model can be installed separately in the following way:
```
python -m spacy download en_core_web_sm
python -m spacy download en_core_web_trf
```

They are then utilized with the spacy nlp pipeline:
https://spacy.io/usage/linguistic-features

OR, if you would prefer to use Java, you can use both these java dependency parse models:
Stanford PCFG Dependency Parser (englishPCFG.ser.gz):
http://nlp.stanford.edu:8080/parser/

Stanford Neural Network Dependency Parser:
https://nlp.stanford.edu/software/nndep.html

Provide dependency parses in your report in the form of dependency trees, which can be hand-drawn or automatically generated (such as with spacy) from the dependency parses produced by both systems.

Software Engineering (includes documentation for your programming assignments)

Your README file must include the following:

- Your name and email address.
- *Homework number* for this class (NLP CS6320), and the *number of the problem* it solves.
- A description of every file for your solution, the programming language used, supporting files, any NLP tools used, etc.
- How your code operates, in detail.
- A description of special features (or limitations) of your code.

Within Code Documentation:

- Methods/functions/procedures should be documented in a meaningful way. This can mean expressive function/variable names as well as explicit documentation.
- Informative method/procedure/function/variable names.
- Efficient implementation
- Don't hardcode variable values, etc