CSC 483/583: Programming Project (200 pts)

The Fake News Challenge: Stance Detection

Due before 11:59 P.M., May 7

For this project you must submit:

- An archive containing all the source code.
- Binaries/jars (if a language that requires compilation is used) and/or instructions how to run the code.
- A PDF document that must contain at least:
  - Description of the command line to run your code, with an example.
  - Description of the code. You don't have to describe every function implemented. But you should describe the main part of the code and indicate where each question is addressed.
  - Results, i.e., the output of your code, for all the questions below that required programming.
  - Answers for all questions that do not require programming.

Answers are always graded by inspecting both code and documentation. Missing code yields no credit. Missing documentation yields partial credit (if code exists and produces correct results).

Because the credit for graduate students adds to more than 200, graduate students' grades will be normalized at the end to be out of 200. For example, if a graduate student obtains 220 points on this project, her final grade will be $220 \times \frac{200}{250} = 176$. Undergraduate students do not have to solve the problems marked "grad students only." If they do, their grades will not be normalized but will be capped at 200. For example, if an undergraduate student obtains 210 points on this project (by getting some credit on the "grad students only" problem), her final grade will be 200.

Important notes:

This project follows the Fake News Challenge (FNC), available at: `http://www.fakenewschallenge.org`. Please read the information on this website in addition of the text provided here!

This project requires machine learning rather than (or in addition of) information retrieval! If you prefer to work solely with information retrieval models, please choose the other project offered ("Building a part of Watson").

TL;DR:

Implement a submission for the FNC using machine learning.

From the FNC website:

"The goal of the **Fake News Challenge** is to explore how artificial intelligence technologies, particularly machine learning and natural language processing, might be leveraged to combat the fake news problem. We believe that these AI technologies hold promise for significantly automating parts of the procedure human fact checkers use today to determine if a story is real or a hoax.

Assessing the veracity of a news story is a complex and cumbersome task, even for trained experts 3. Fortunately, the process can be broken down into steps or stages. A helpful first step towards identifying fake news is to understand what other news organizations are saying about the topic. We believe automating this process, called **Stance Detection**, could serve as a useful building block in an AI-assisted fact-checking pipeline. So stage #1 of the Fake News Challenge (FNC-1) focuses on the task of Stance Detection."

Task Definition:

Input:
A headline and a body text – either from the same news article or from two different articles.

Output:
Classify the stance of the body text relative to the claim made in the headline into one of four categories:

- *Agrees*: The body text agrees with the headline.
- *Disagrees*: The body text disagrees with the headline.
- *Discusses*: The body text discuss the same topic as the headline, but does not take a position
- *Unrelated*: The body text discusses a different topic than the headline

See the FNC for examples for these four classes, as well as for examples and descriptions of the actual data.

Please note that the FNC provides the official testing dataset on June 1st, which, unfortunately, is after our class ends. For this reason, we will *not* be using the official datasets for this project. Instead, the instructor will partition the official training dataset into two dataset: our own, local training dataset, and our own (smaller) testing dataset. Please see D2L for these data.

Your project should address the following points:

1) (125 pts) Classification using machine learning: Implement a machine learning (ML) classification system using Support Vector Machines (SVM) for the four classes above. I recommend one of these packages for the ML part:

- For C/C++: `http://svmlight.joachims.org`
- For Python: `http://scikit-learn.org/stable/`
- For Java: `https://www.csie.ntu.edu.tw/~cjlin/liblinear/`
- For Scala: `https://github.com/clulab/processors`

Describe how you pre-processed the terms in the headline and document (stemming, lemmatization, stop words, etc.). Describe what features did you create for your classifier. For example, how did you model the overlap between the headline and the body of the article? How did you model negation? Describe how you implemented the ML component. Note that you are allowed to look at the baseline provided by the FNC organizers, but you are not allowed to use their code. You have to implement your own classifier.

2) (25 pts) Measuring performance: Measure the performance of your FNC system, using the official FNC evaluation measure (see the FNC website for details). Discuss how this measure compares to standard accuracy. Make sure your report includes evaluation scores *on our testing dataset*.

4) (50 pts) Error analysis: Perform an error analysis of your best system. Sample several stances that your system classified incorrectly (say 100), and try to identify error classes in this subset. For example, how many stances were classified poorly due to your incomplete handling of negation? How many were classified incorrectly due to the fact that your system (probably) does not address synonyms and antonyms? Discuss the error classes you found, and include a histogram of these error classes from your analysis (that is, how many questions appear in each class).

Lastly, what is the impact of stemming and lemmatization on your system? That is, what is your best configuration: (a) no stemming or lemmatization; (b) stemming, or (c) lemmatization? Why?

5) (50 pts) Improving classification (GRAD STUDENTS ONLY): Improve the above standard stance classification system using natural language processing and/or machine learning. For this task you have more freedom in choosing a solution and I encourage you to use your imagination. For example, you could use WordNet to capture and handle synonyms and antonyms. You could use the simple idea from this paper to handle negation (see the paragraph starting with "One unconventional

step. . . ” in: https://arxiv.org/abs/cs/0205070). You can switch to a more complex learning algorithm such as a neural network.