CS 343: Final Project

Francisco Iacobelli
November 16, 2023

1 TASK

Create a program that will perform a natural language processing task. This task will require you to train some sort of language model and then load that model, input text, and produce some output.

Some possible projects are:

- A question–answering system.
- A text summarization system.
- A Part of Speech Tagger that detects when sentences are incomplete and asks the user for more information on the subject.
- A Relationship finder (answers the question "what is a X")
- A synonym finder
- A paraphraser
- A tutoring system. (Asks questions and tells you what's missing in your answer)
- A system to summarize data
- Another system that performs some understanding task.
- Something using ChatGPT/Bard/etc. where you supplement some of the information to achieve a specific goal.

Each of these projects can have many variations. For example A question answering system can use an information-retrieval approach, a knowledge base approach or a mixed approach. It can include new techniques you think apply. In addition, you can use a question – answering system for specific texts (insurance policies, univeristy data, medical discharge instructions, etc.).

Another exaple is the text summarization system. You can create a generic text summarization, a newspaper text summarizer, etc. Or summarize tweets with a given hashtag. etc. Another thing you can do is to have a question answering system where you use chatGPT where you have answers and ask it to formulate a question for that answer and you use similarity metrics to judge the answer (or chatGPT in conjunction with your answer.)

1.1 Notes

- You can use any libraries you want or any APIs. Most these systems have libraries to help with. You can look for a tutorial online, but please read the **Criteria** section for clarification on how to deal with tutorials online.
- It is very important that youseek me for a one-on-one meeting **before** you get your grade for the **Project Check-in** assignemnt.
- Your program can be split into more than one python file. For example, some people like to have a python file for training and another that interacts with the user. Others like to have it all in one program. This is your choice.
- You must create a README.txt file with instructions on how to run your program and it must run in my computer. So... **test, test, and test it!**
- As a sub note of the above: Your README.txt should be very precise. A `README.txt` with instructions like "run the main file" are not acceptable. See example below.

- Your `README.txt` file should have at least three sections: `Libraries` detailing the libraries you use; `Running` detailing how to run the program; and `Testing` detailing how to test the system.
- You also have to have a TEST.txt file that contains 10 tests for your program. These can be automatic tests or manual tests. It is up to you.

2 Example

This example uses one of your previous homework assignemnts to illustrate what a project would look like. Let's say I create a part of speech tagger. It is comprised of two files `train.py` and `tagger.py`. Let's say I use the `ca_train.txt` file to train my tagger.

- the `README.txt` file may contain the following:

```
Libraries:
This program uses: nltk
You must also install numpy

Running the program:
To run this program you must execute:
python train.py

This program will create a file with counts of bigrams
and trigrams and unigrams called out.txt

Then run:
 python tagger.py
The program will read the file out.txt automatically and ask you
to enter a sentence. Enter any sentence.
The program will then output each word of the sentence and its tag.

Testing:
To test the program, look into TEST.txt. Each line is a test sentence.
After the period is a sequence of tags. Those are the expected tags.
If you copy and paste the sentences at the prompt of tagger.py.
Then, tagger.py should produce the expected tags, or tags that are
very close to those. (80\% accurate).
```

- The `TEST.txt` Should have 10 test cases. The first two lines of the file may look like:

```
the cat in the hat knows a lot about that. dt nn pp dt nn vb dt ad pp dt
jack and jill went up the hill. nn cj nn vbd vbd-pp dt nn
```

Alternatively you can create a third program that analyzes the sentences in the test file and outputs the tags. Simply specify that in the `README.txt` file.

Also, if you are creating some sort of dialogue system, you may have each line of `TEST.txt` be a dialogue, and the next line be the expected output of the system. You can create some 10 interactions that way.

3 CRITERIA

- If the program is submitted as specified: 20pts.
- If the program runs 30pts. (that is, I can run it using your README.txt file and there are no infinite loops, hangups or no uncaught errors of any kind)
- It works as described: 50 pts. (I will use your test files and my own testing)

- If you use code from a tutorial or other external source you must put, in comments, the source of the code (website, chatGPT, chegg, etc.). If more than 30% of your code comes from external sources, you will receive a zero for the project.
  - This means that you can use a tutorial for reference (and perhaps there is not much you can do in terms of the core algorithms), but you can use your own corpus and create your own interaction and output. That would take care of some 70% of the code.
  - Your test program,if present, also counts towards this code.
  - Code covered in class does not count as external source.

4 SUBMIT

One ZIP (not arj, rar, etc.) file with all the necessary files to run the program and two extra files: (a) A text file called `README.txt` (Yes, README is one word, all caps) with the instructions to run your program. If you need to install special libraries in python, mention that on the README file; and (b) a text file called `TESTS.txt` with test cases or commands to test your program. You must have at least 10 test cases.

I advise you test these instructions with a friend or in someone's computer. If I cannot run it, I cannot grade it. If I cannot grade it, you may receive a zero because these will be graded very close to the end of the semester.

Lastly, if you are going to use a library, you are welcome to let me know so I can install it in advance, or put that in the README file.