CSE583: Natural Language Processing

Zhan Shi, Neilly Herrera Tan, Angel Burr

Procedural specification

- **Background**

  Natural language processing affords new possibilities for sentiment analysis of text. This is especially beneficial in everyday scenarios when interacting with any chatbot: for example, when asking Siri for weather updates or interfacing with a customer service bot.

  To this end, we seek to investigate how to better understand natural language processing. Our main problem being addressed is how to generate fake text that effectively matches the style of an input corpus. This program will specifically generate novel poems based on trained datasets of poems by different authors.

- **User profile**
  - Someone who wants to generate any fake text and who wants to know a little bit more about natural language processing. This user will have a basic knowledge of Python, and is able to run the program using CLI.

- **Data sources**

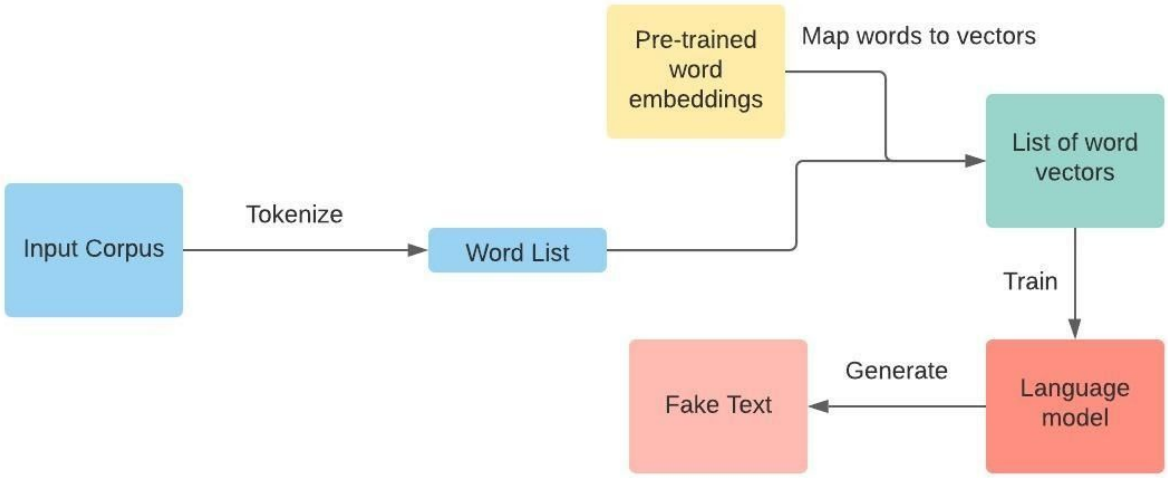
  We will use input data from a collection of poems via Project Gutenberg, on their poetry bookshelf: http://www.gutenberg.org/ebooks/bookshelf/60
  - Poems by TS Eliot: http://www.gutenberg.org/cache/epub/1567/pg1567.txt
  - Poems by John Keats: http://www.gutenberg.org/cache/epub/2490/pg2490.txt
  - Poems by Emily Dickinson: http://www.gutenberg.org/cache/epub/2678/pg2678.txt

- **Use case**

  The main use case for this program is for generating fake text. Our program takes an input of some text, and outputs a wall of fake text with similar sentiment to the input text.
  - User input: A few paragraphs of text such as poems from various authors.
  - Output: Computer generated fake text that mimics the style of the input text.

  This text generator program is meant for fun. With a more advanced program, a person may be able to generate greeting cards or fortune cookies based on larger or smaller text inputs. These examples are implications for future work. For class purposes, our program may not be advanced enough to generate meaningful text in this manner.

Component specification



- **Preprocessing**
  Preprocess the input text with 4 main functions in mind:
  1. Tokenize the input.
  2. Indexing
  3. Generate <EndOfSentence> tokens
  4. Break token list into pieces based on <EOS> tokens for batch training

- **Training**
  Train the language model with the input given, containing two parts:
  1. Initializing the embedding matrix with pre-trained vectors when available, otherwise start at random.
  2. Train the LSTM model

- **Decoding**
  Decode the language model to generate fake text. With both greedy method and top k random sampling method in mind.