CS 505 – Spring 2021 – Assignment 2 (100 pts, bonus: 10 pts) – Scraping, Text Processing, LM, Analysis

Problems due 11:59PM EST, February 26.

In this assignment, you will learn all about scraping, pre-processing, and conducting preliminary analysis of text, which are very important when doing NLP, and use python libraries such as NLTK, spacy, which are popular in NLP. You have 2 weeks to finish this particular assignment.

Submit in Blackboard by 11:59PM EST, February **26.**
–Please indicate names of those you collaborate with.
–Every late day will reduce your score by 20
–After 2 days (i.e., if you submit on the 3rd day after due date), it will be marked 0.

Submit your (1) code, (2) README.txt containing the instruction to run your code, (3) extracted data (tweets' json, Wikipedia text, and news text), and (4) write up in one zip file.

When necessary, you must show how you derive your answer

Problem 1. *(15 pts) In online discussion forums, such as Reddit, discussions are broken down into different communities. Given such forum:*

1. *(5 pts) How do you determine which community a post is likely from in an unsupervised manner?*
2. *(5 pts) How can you automatically generate posts that will fit a particular community?*
3. *(5 pts) If there is a debate inside a particular community regarding a specific topic, say COVID-19, and given that the points of contentions come from this list: mask wearing, reopening, vaccination; how do you determine which stance a person is taking in a post about COVID-19?*

For each of the task above, please specify what type of model you can use to address the task and identify what would be the training data, features, and labels (if any), and what would be the output of the model.

Problem 2. *(5 pts) Use maximum likelihood estimate to derive unigram $P(w_i)$, bigram $P(w_i|w_{i-1})$, trigram $P(w_i|w_{i-1}, w_{i-2})$ probabilities i.e., slide 16 in Language Model lecture.*

Problem 3. *(5 pts) In the Language Model lecture (slide 35), we derive the formulation of perplexity of a single test sentence. Derive the formulation of perplexity for the whole test set containing k sentences for a trigram language model.*

Problem 4. *(25 pts, bonus: 10 pts)* ***Twitter Scraping.*** *Use your Twitter Developer API to scrape 10,000 most recent tweets in the English language from Twitter with the keyword 'covid'. You can use the search function of library such as* ***Twython****. Out of these 10,000 tweets, use 9,000 to train a unigram, bigram, and trigram language models (LMs). Use* ***NLTK*** *library with KneserNeyInterpolated language model (currently possibly the best for smoothing) to build your LMs to deal with zero-count ngrams. Remember to process the text first before using it to train your LMs i.e., sentence segmenting, tokenizing, lower casing, and padding with begin-of-sentence and end-of-sentence symbols (all of these can be done within NLTK). Use the same pre-processing on your test text.*

1. *(9 pts) Report the average perplexities of your language models on the remaining 1,000 tweets i.e., use NLTK LM perplexity function to compute the perplexity of each tweet, and then average.*
   Note that NLTK is implementing perplexity slightly differently than what we discuss in class with regards to normalizing, it normalizes based on the number of ngrams instead of the length of the sentence—you should see the source code of NLTK to find out more.

2. *(6 pts) Generate 10 tweets using each of your language model (for a total of 30 tweets). For each language model, mention interesting observation from its generated tweets e.g., are they coherent? do the tweets reflect interesting topics?*

3. *(10 pts, bonus: 10 pts) Using NLTK library (with VADER, which is a lexicon and rule-based sentiment analysis model), compute the sentiment of each tweet in all your 10,000 tweets.*

   (a) *(4 pts) What is the average **compound** sentiment of the tweets from VADER? Are users in your collected tweets generally positive/neutral/negative when talking about COVID-19?*

   (b) *(6 pts) After removing stopwords using NLTK, for positive tweets, what are the top 10 words mentioned? and for negative tweets, what are the top 10 words mentioned?*

   (c) *(Bonus 10 pts) Using only tweets that are geo-located with **country_code** US i.e., has non-null child object **place** in its json, extract the state information from the **full_name** child object of place. Report average sentiment compound scores from each of the state you found. Which state in your data has the most positive users, which state has the most negative users?*

Problem 5. *(30 pts) **Wikipedia Scraping.** Use library such as **requests** to scrape HTML of this article in Wikipedia: https://en.wikipedia.org/wiki/COVID-19_pandemic and scrape also the HTML of articles within Wikipedia that are linked from only the **content** of this page i.e., you don't need to scrape the sidebar—you will have to look at the retrieved HTML of the first page and see the pattern you can use to obtain links from this article's content to other Wikipedia articles. See https://en.wikipedia.org/wiki/Wikipedia:What_is_an_article%3F for what is defined as an article in Wikipedia.*
*Once you retrieve all the articles, using library such as **BeautifulSoup** or regular expressions of your creation, extract only the text of each article's content.*

1. *(10 pts) Sentence split, tokenize, lemmatize, lower case, then remove stop words from the text using the library **spacy**. Then, construct a vocabulary of words in the text.*

   (a) *(5 pts) What are the top 20 words in the vocabulary according to frequency? Are they from a specific topic? Do they give you insights into what the text is all about?*

   (b) *(5 pts) Using library such as **wordcloud**, generate the word cloud of the text to visualize the distribution of words—include the word cloud image in your write up. Does the word cloud give you some insights into what the text is all about?*

2. *(10 pts) Sentence split, tokenize, lemmatize, lower case, then remove stop words from your 1,000 test tweets from **Problem 4** using spacy.*

   (a) *(2 pts) Compute how many word types in your tweets are out-of-vocabulary, normalized by the number of word types in your tweets, when using vocabulary constructed from Wikipedia above.*

   (b) *(2 pts) Compute how many word tokens in your tweets are out of vocabulary, normalized by the number of word tokens in your tweets. This is the OOV-rate of your tweet test set.*

   (c) *(4 pts) Compute the OOV-rate of your tweet test set when using your 9,000 train tweets from **Problem 4** to construct your vocabulary/lexicon. Note that you have to do the same pre-processing on your tweet train set (i.e., sentence split, tokenize, lemmatize, lower case, then remove stop words using spacy) before constructing the vocabulary.*

(d) *(2 pts) What does the OOV-rate tell you about the domain of these two texts (Wikipedia vs. Twitter of similar topic that is COVID-19)?*

3. *(10 pts) Sentence split, tokenize, and lower case the Wikipedia data you have collected, then get the first 9,000 sentences from the data—most of the sentences therefore will come from the first URL that you scrape: https://en.wikipedia.org/wiki/COVID-19_pandemic. Then, train a trigram KneserNeyInterpolated language model based on these 9,000 sentences (remember to pad with begin- and end-of-sentence symbols).*

   (a) *(5 pts) Report the average perplexity of the model on your Twitter test sentences, the one that contains 1,000 tweets from* ***Problem 4*** *(remember to pre-process the test set the same way you pre-process the training data of your LM).*

   (b) *(5 pts) Compare this perplexity to the one you obtain in Problem 4.1 for the trigram LM trained on tweets. What does the perplexity difference tell you about the domain of these two texts (Wikipedia vs. Twitter of similar topic that is COVID-19)?*

Problem 6. *(20 pts)* ***News Scraping.*** *Scrape ABC[1] and Fox News[2] articles from their sitemaps (aim for at least 100 articles from each site – you can get more). You can use this github project: https://github.com/pmyteh/RISJbot for scraping, or you can build your own (e.g., using the library* ***newspaper*** *can allow you to obtain full text of news articles given their URLs). Extract the text of the articles, then sentence split, tokenize, and remove stop words using* ***spacy****.*

1. *(10 pts) Construct word type-word token graph of news texts from these two news sites, where x-axis is #token, and y-axis is #type. As the number of tokens grow, the number of word types would grow and then plateau at some point. Include the type-token graph in your write up. Do you see interesting insights when comparing the two graphs?*
2. *(10 pts) Construct the word clouds from the two texts. Include the word clouds and interesting insights from them in your write up.*

[1] https://abcnews.go.com/xmlLatestStories

[2] https://www.foxnews.com/sitemap.xml?type=news