CS 505 – Spring 2022 – Assignment 1 (100 pts, bonus: 10 pts) – Knowledge about Language (syntax, lexical semantic, morphology), Scraping, Text Processing, LM, Analysis

Problems due 11:59PM EST, February 21.

In this assignment, you will learn about linguistic knowledge (syntax, lexical semantic, morphology), 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 Gradescope by 11:59PM EST, February **21.**
–Download the submission (answer sheet) template at this **link** and use it to write down your write-up answers.
–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 (Jupyter Notebook), (2) extracted data (tweets' json, Wikipedia text, and news text), and (3) submission (answer sheet) template in one zip file.

When necessary, you must show how you derive your answer

Problem 1. *(5 pts) In online discussion forums, such as Reddit, discussions are broken down into different communities. Given such forum and using your knowledge of language modeling so far:*

1. *(2.5 pts) For a given post, how do use language model to determine which community it likely belongs to?*
2. *(2.5 pts) How do you automatically generate posts that will fit a particular community using language modeling?*

Problem 2. *(20 pts) In this problem, you will learn to think like a computer and understand and translate a language that is unknown to you (Tshiluba) to/from a language that is known to you (English). To help you along, here are some sentences in Tshiluba and their corresponding English sentences. You must derive knowledge of the language such as morphology and lexical semantics from these examples to help you answer the following questions (we humans are very good at this and do not need thousands of sentences to derive these knowledge).*

mukaji uvwa mumona muana → the woman saw the child
bakaji bavwa bamona muana → the women saw the child.
muluma uvwa mumona bakaji → the man saw the women.
muluma uvwa mumona bambuji → the man saw the goats.
muluma udi mumona bambuji → the man sees the goats.
banzolu bavwa bamona bantambwe → the chickens saw the lions.
tubambwa tuvwa tumona baluma → the small dogs saw the men.
mbwa uvwa mumona ntambwe → the dog saw the lion.
ntambwe uvwa mumona tubanzolu → the lion saw the small chickens.
kanzolu kavwa kamona tubantambwe → the small chicken saw the small lions.
tubakulu tuvwa tumona mbwa → the small adults saw the dog.

kamuntu kavwa kapeta ntambwe $\rightarrow$ *the small person found the lion*
kamulunda kavwa kapeta ntambwe $\rightarrow$ *the small friend found the lion*
fruit $\rightarrow$ *cimuma*
fruits $\rightarrow$ *bimuma*

1. *Translate the following into Tshiluba:*
   (a) *(2 points) the men found the children*
   (b) *(2 points) the chicken saw the dogs*
   (c) *(2 points) the small chickens found the goat*
   (d) *(2 points) the small child found the small goat*
   (e) *(2 points) the friend found the goat*
2. *translate the following into English:*
   (a) *(2 points) bantu*
   (b) *(2 points) kamukulu*
   (c) *(2 points) tubalunda*
   (d) *(2 points) cimuma civwa cimona kantambwe*
   (e) *(2 points) kantambwe kavwa kamona tubimuma*

Problem 3. *(25 pts, bonus: 10 pts)* ***Twitter Scraping and pre-processing using NLTK.*** *Use your Twitter Developer API to scrape 10,000 most recent tweets in the English language from Twitter with the keyword 'nft'. Additionally scrape 1000 tweets using the keyword 'football'. Save your tweets in a two files with respective names for submission. You can use the search function of library such as* ***Twython****.*

Out of the 10,000 nft 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. This particular language model can deal with zero-count ngrams(out of vocabulary ngrams).*

Pre-process the text first before using it to train your LMs. The pre-processing steps that you must do are:

- *sentence segment/split*
- *tokenize (you should use TweetTokenizer from NLTK.tokenize)*
- *lower case*
- *padding with begin-of-sentence and end-of-sentence symbols (check out steps how to do this and train NLTK LM at this* ***link****)*

all of these can be done within NLTK. ***Don't forget*** *to do the same pre-processing on your test examples before processing the with the LM.*

1. *(5 pts) Report the average perplexities of your language models on the remaining 1,000 tweets with the keyword 'nft' i.e., use NLTK LM perplexity function to compute the perplexity of each tweet, and then average.*
2. *(4 pts) Report the average perplexities of your language models on the test set that you scraped using the keyword 'football'. How are the perplexities changed?*

3. *(6 pts) Generate 10 tweets using each of your language models (unigram, bigram and trigram models) (for a total of 30 tweets).*

4. *(10 pts, bonus: 10 pts) Using NLTK library (with VADER, which is a lexicon and rule-based sentiment analysis model, see this* ***link*** *for example usage), 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 NFTs?*

   (b) *(6 pts) For positive tweets, what are the top 10 non stop words mentioned? Similarly, for negative tweets, what are the top 10 non stop words mentioned? (See this* ***link*** *for an example of removing stop words from a list using NLTK)*

   (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 4. *(30 pts)* ***Wikipedia Scraping and pre-processing using Spacy.*** *Use library such as* ***requests*** *to scrape HTML of this article in Wikipedia:*
`https://en.wikipedia.org/wiki/Non-fungible_token`.

Then, scrape the URLs of all 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 the links to all the articles (include the original article `https://en.wikipedia.org/wiki/Non-fungible_token`*), using library such as* ***BeautifulSoup*** *or regular expressions of your creation, extract only the text of each article's content. Save your Wikipedia texts in a single text file for submission (one line per Wikipedia article).*

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

   (a) *(5 pts) What are the top 20 non stop words in the vocabulary according to frequency (use* ***Spacy*** *to remove stop words like shown in this* ***link****)? 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 non stop words—include the word cloud image in your submission (answer sheet) template. Does the word cloud give you some insights into what the text is all about?*

2. *(10 pts) Use* ***Spacy*** *again to sentence split, tokenize, lemmatize, lower case the 1,000 non-preprocessed test tweets with keyword 'nft' you've collected in* ***Problem 3****.*

   (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 texts 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 3*** *to construct your vocabulary/lexicon. Note that you have to do the same pre-processing on your non-processed tweet train set (i.e., sentence split, tokenize, lemmatize, lower case 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 NFT)?*

3. *(10 pts) Get the first 9,000* ***sentences*** *from the processed Wikipedia data from* ***Problem 4.1****—most of the sentences therefore will come from the content of the first article that you scrape:* `https://en.wikipedia.org/wiki/Non-fungible_token`. *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 this Wikipedia-trained language model on your processed Twitter test sentences from* ***Problem 4.2****, (remember that we have used the same pre-processing steps for this test set as the pre-processing we did on the Wikipedia training data for the LM).*

   (b) *(5 pts) Compare this perplexity to the one you obtain in* ***Problem 3.1*** *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 NFT)?*

Problem 5. *(20 pts)* ***News Scraping and pre-processing using Spacy.*** *Scrape ABC*[1] *and Fox News*[2] *articles from their sitemaps (aim for at least 100 articles from each site – you can get different lists of articles by refreshing the sitemaps). You can use this github* ***project*** *for scraping, or you can make your own (i.e., use regular expressions to extract news URLs from the sitemaps, then use libraries such as* ***newspaper*** *to obtain full text of news articles given their URLs). Save the news texts into a single text file for submission (one line per news article).*

Pre-process the texts of the articles by doing these steps: sentence split, tokenize, lemmatize, lower case, and remove stop words using ***Spacy****.*

1. *(10 pts) Construct a histogram of word count from both sources. The X-axis should be unique words in decending order of count and the Y-axis should be the counts for each word. Compare the two sources, which curve goes down faster? Which has a more diverse vocabulary? Are the dominant words similar or different?*

2. *(10 pts) Construct the word clouds from the two texts. Include the word clouds and interesting insights from them in your submission (answer sheet) template.*

Useful Links, These links are also present in the homework where they are relevant as hyperlink. They are also added here so no one will miss them.

- Answer sheet template:
  https://docs.google.com/document/d/1PTggKhoTK82h-jQYsZIjSq9kV9DMtG38cEybbXsAhh0/edit?usp=sharing
- Training NLTK language model: https://www.kaggle.com/alvations/n-gram-language-model-with-nltk
- Sentiment analysis with VADER: https://www.geeksforgeeks.org/python-sentiment-analysis-using-vader/
- Remove stop words using NLTK: https://www.geeksforgeeks.org/removing-stop-words-nltk-python/
- Remove stop words using Spacy: https://stackabuse.com/removing-stop-words-from-strings-in-python/
- News scraping project: https://github.com/pmyteh/RISJbot

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