CMP711 Natural Language Processing

HW01 – Basic Text Processing & Language Models

Due Date: November 1, 2024 (Friday)

You have to write a Python program (you may use NLP libraries such as **nltk**) which will perform the following tasks:

- Your program should read a name of an input text file (.txt file) which contains a Turkish text in UTF-8 format. Then it should read the content of that file.
- Your program should find the number of the sentences of the input text file and print the number of the sentences in the text. In order to find the end of the sentences, your program should assume that the sentences end with symbols *dot*, *question mark* or *exclamation mark.*
- Your program should tokenize the text in order to create a token list. A token is a word or an end of sentence symbol (dot, question mark or exclamation mark). A word is a string of Turkish and English letters. In order to normalize words, all words should be converted to lowercase. Then your program has to remove all end of sentence marks from the token list, and put <s> token into the beginning of the sentences and </s> token into the end of sentences. Thus, your token list will also contain tokens <s> and </s>.
- Your program should find unigram and bigram counts (frequencies) for the text and their probability values.
- In the **first step**, your program should print the following information into a text file (**result.txt**):
  - The number of sentences in the text.
  - The total number of tokens in the text (Corpus Size).
  - The number of unique words in the text (Vocabulary Size).
  - Unigrams: Your program should print all sorted unigrams (sorted wrt their frequencies in descending order) together with their frequencies and their probability values.
  - Bigrams: Your program should print all sorted bigrams (sorted wrt their frequencies in descending order) together with their frequencies and their probability values. You should only print bigrams with non-zero frequencies.
- In the **second step**, your program should replace the token with the lowest frequency with the token UNK. Then it should compute the smoothed bigram probabilities using Add-k smoothing method (assume that k=0.5). After finding smoothed probability values of all bigrams (including the bigrams containing UNK), your program should print the following information into **result.txt** file
  - Smoothed Bigrams: Your program should print top 100 sorted smoothed bigrams (sorted wrt their smoothed probabilities in descending order) together with their original probability values and their smoothed probability values.
- In the **last step**, your program should read at least two sentences and compute their probabilities using the smoothed bigram probability values. If a sentence contains a word which is not in the text, your program should use the probability values of UNK token for that word. Then it should print those sentences into **result.txt** file together with their probabilities.

You should test your program with at least given two sample files (`hw01_tinytr.txt`, `hw01_bilgisayar.txt`). You will submit the produced result files for each file together with your program. You should create a result file for each test file (such as `hw01_tinytr_Result.txt` and `hw01_bilgisayar_Result.txt`) to submit together with your program. **The content of each result file should be as follows:**

Number of Sentences in File: ....

Number of Total Tokens (Corpus Size): ....

Number of Unique Words (Vocabulary Size): ....

Unigrams Sorted wrt Frequencies (from Higher to Lower Frequencies):

Unigram1 ItsFrequency ItsProbability

Unigram2 ItsFrequency ItsProbability

...

Bigrams Sorted wrt Frequencies (from Higher to Lower Frequencies):

Bigram1 ItsFrequency ItsProbability

Bigram2 ItsFrequency ItsProbability

...

After UNK addition and Smoothing Operations,

Top 100 Bigrams wrt SmoothedProbability (from Higher to Lower):

Bigram1 ItsSmoothedProbability ItsProbability

Bigram2 ItsSmoothedProbability ItsProbability

...

SampleSentence1 ItsComputedProbability

SampleSentence2 ItsComputedProbability

Hand in:

You will submit your homework using the HADI system. You have to upload the following three files.

- The source code (a Python program) of your homework. Put the source code of your program into a **.txt** file (**`hw01.txt`**) and upload this .txt file. **Make sure that this file contains only your Python program.**
- The result file (**`hw01_tinytr_Result.txt`**) containing the results of the file **`hw01_tinytr.txt`** and the probability of at least two sentences.
- The result file (**`hw01_bilgisayar_Result.txt`**) containing the results of the file **`hw01_bilgisayar.txt`** and the probability of at least two sentences.