Tarea 2 – ISIS 4221

Natural Language Processing

Due date: 18-09-2024 6pm

Coding rules: Use jupyter notebooks and be sure that the notebook is executed and contain the results before submitting. All classes, methods, functions and free-code MUST contains docstrings with a detail explanation.

Report: Together with the notebooks, you must submit a written report (please use pdf format) with the answers to the questions and a short summary of the implementation.

Submission: Assignments are submitted via Bloque Neon. Do not email us your assignments. Please upload all files and documents. You can work in pairs or individually.

Datasets

- **20N: 20Newsgroups (http://qwone.com/~jason/20Newsgroups/)**
- **BAC: The Blog Authorship Corpus (https://huggingface.co/datasets/blog_authorship_corpus)**

PLEASE READ DATASET DESCRIPTIONS

You can download all datasets from:

https://www.dropbox.com/sh/pzukl8aztgecvio/AAAqHBPfpH8lqQLOqbIGVfOHa?dl=0

N-Gram Language Models Implementation

For the 20N and BAC datasets, perform the processing required to build two N-Gram Language Models:

I. (5p) Read the files and build two large consolidate files that are the union of all the documents in 20N and BAC.

II. (5p) Tokenize by sentence.
   - Normalize, but DO NOT eliminate stop words.
   - Replace numbers with a token named NUM.
   - Add sentence start and end tags <s></s>.
   - Tokens with unit frequency should be modeled as <UNK>.

III. (10p) Select 80% of the resulting sentences -random without replacement- to build the N-gram model and the remaining 20% for evaluation. Create the following files:
   - **20N_<group_code>_training (training sentences)**
   - **20N_<group_code>_testing (testing sentences)**
   - **BAC_<group_code>_training (training sentences)**
   - **BAC_<group_code>_testing (testing sentences)**

IV. (50p) Build the following N-gram models using Laplace smoothing and generate an output file for each one (you choose the output structure, but be sure to provide an appropriate python reading method/function):

- **20N_<group_code>_unigrams**
- **20N_<group_code>_bigrams**
- **20N_<group_code>_trigrams**
- **BAC_<group_code>_unigrams**
- **BAC_<group_code>_bigrams**
- **BAC_<group_code>_trigrams**

V. (15p) Using the test dataset, calculate the perplexity of each of the language models. Report the results obtained. If you experience variable overflow, use probabilities in log space.

VI. (15p) Using your best language model, build a method/function that automatically generates sentences by receiving the first word of a sentence as input. Take different tests and document them.

You must deliver two notebooks. The first must contain the construction process of the language models (literals I, II, III, and IV). The second is the use of the models (literals V and VI). If you generate large files (literal IV) it may take time to upload them to Boque Neon, alternatively you can upload the files to a repository (onedrive, google drive, dropbox, etc) and share them publicly (i.e. via links) in the document so that they can be downloaded.