Homework 1: Exploring Textual Similarity in Documents

Group 45: Ashraf Ahmed

Introduction

This assignment focuses on the practical implementation of identifying textually similar documents through the utilization of Jaccard similarity. The approach involves employing shingling, min hashing, and locality-sensitive hashing (LSH) techniques along with their corresponding algorithms.

Brief Description

We conducted our homework using Python within a Jupyter notebook, aiming to implement and compare shingling and min hashing techniques. The primary emphasis was on testing and evaluating the scalability of our implementation concerning execution time relative to the size of the input dataset.

1. **Data Preparation**
    - We loaded and cleaned the dataset titled "Exploring Climate Change Narratives On Television News 2009-2020" from GDELT Project.
2. **Shingling Implementation**
    - Developed a class named Shingler, which constructs k-shingles of a specified length (e.g., 10) from a given document.
    - The class computes a hash value for each unique shingle and represents the document as an ordered set of its hashed k-shingles.
3. **Jaccard Similarity Calculation**
    - Created a class named CompareSets to compute the Jaccard similarity of two sets of integers, specifically two sets of hashed shingles.
4. **MinHashing Implementation**
    - Implemented a class named MinHashing, responsible for building a MinHash signature (a set of hashed shingles) using K random hash functions. This results in a MinHash signature with K values for each set.
5. **Jaccard Similarity Estimation with MinHash**
    - Developed a class named CompareSignatures to estimate the Jaccard similarity between two sets by comparing their MinHash signatures. The fraction of hash functions for which the signatures match provides an estimate of the Jaccard similarity.
6. **Evaluation Functions**
    - Implemented functions to facilitate the evaluation of our implementations, specifically comparing shingles versus minhash.
    - Sample outputs showcasing the implementation's scalability, including execution time versus the size of the input dataset, are provided in the notebook.

Evaluation Functions:

- **list_shingles(txt_files, shingle_size_k_value)**: Returns a list of all shingles from the input of n documents using the Shingler class.
- **calculate_jaccard_similarity_shingles(shingles_list)**: Computes the similarity matrix between n documents using the Shingler class and CompareSets class.

- **calculate_all_signature(no_hash_functions, shingles_list)**: Generates MinHash signatures for all sets using a specified number of hash functions.
- **estimate_jaccard_similarity(signatures_all, no_hash_functions, shingles_list)**: Estimates Jaccard similarity by comparing MinHash signatures.