CS 470 Homework 5

Due by Thursday, April 6, 2023 at 6:00 PM

Submission instructions

Submit your assignment through the QTest system, using exam ID: **CS470.hw5**. Upload a single ZIP archive file named **hw5.zip**, containing all the files of your solution:

1. The program's source files.
2. A `README.txt` file explaining how to compile and run your program.
3. The test document files.
4. The report in PDF format.
5. The LaTeX source files used to typeset the report.

No email submissions are accepted. No late submissions are accepted.

At the top of your solution, include a section named "Collaboration statement" in which you acknowledge any collaboration, help, or resource you used or consulted to complete this assignment.

1 MinHash Similarity (70 points)

Read the blog post of Chris McCormick about his implementation of the MinHash technique for comparing text documents:

https://chrisjmccormick.wordpress.com/2015/06/12/minhash-tutorial-with-python-code

Then, download and study his implementation from GitHub:

https://github.com/chrisjmccormick/MinHash

As you read through the blog post and the code, compare it with what you learned from studying Chapter 3 of the "Mining of Massive Datasets" (MMDS) textboox (http://www.mmds.org).

Then, choose one of the following two options:

1.1 Option 1 (for Java programmers): Pyhton to Java conversion

Chris's implementation is written in Python. Translate this program in standard Java. If some of the Python functionalities used in the program is not directly available in the Java standard library, choose an appropriate replacement that would achieve the same goals. Explain and justify your choices in the report.

1.2 Option 2 (for Python programmers): Banding technique extension

As a pre-processing step, convert the original implementation (which is written in Python 2) to the more modern Python 3. This conversion is fairly straightforward and requires only minimal changes to the original code.

Then, observe that in this code the size of the minhash signatures is set to a quite small value: `numHashes = 10` (line 47 of file `runMinHashExample.py`). This choice makes the pairwise comparison between all the minhashes very fast (at least for the provided dataset), but for a larger dataset such a small minhash size could result in an inaccurate representation of the original documents. Setting a larger minhash size (for example, `numHashes = 100`) makes the signatures more accurate, but the comparison between them becomes much slower (try it). To solve this problem, the MMDS textbook describes a "banding technique" to perform locality-sensitive hashing of minhash signatures (section 3.4). Extend the implementation using this thechnique. Then, run some experiments with varying mihash sizes and describe your results in the report.

2 Report (30 points)

Write a report in LaTeX to present your results. Describe your implementation and your experiments. Discuss your results, the insights, experiences and lessons you have learned from this assignment.

Grading criteria

- 70 points for correct and complete translation or extension of the MinHash code (depending on the chosen option). -5 for minor mistakes; -10 if there is some mistake but the program still works in most cases; -20 for more serious mistakes but the program still works in several cases. Zero points if the program does not compile, or if the program compiles but gives mostly the wrong results or crashes.
- 30 points for a complete, clear, and well organized report. Zero points if the report is not typeset using LaTeX, or if the LaTeX source code is not provided.
- -10 points for insufficient comments in the code.
- -10 points for each deviation from the submission instructions.
- -10 points for missing collaboration statement.