Com S 435/535: Large Scale Dataset
Report: Programming Assignment 2

Authors: Yijia Huang and Mohammad Wardat

---

3. MinHash

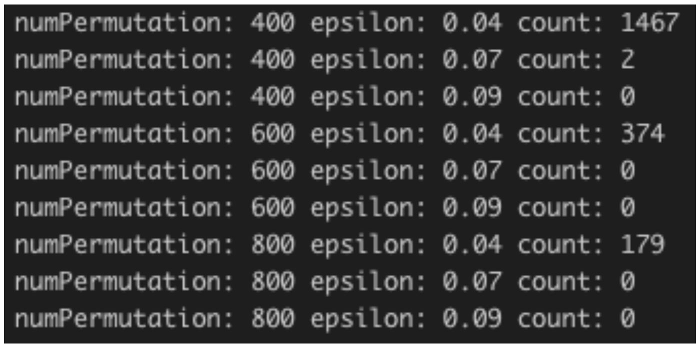
- Your procedure to collect all terms of the documents and the data structure used for this.
    - **Data Structure**
        - We used a HashMap<String, Integer> to store all terms over all the documents, where the terms as keys and the assigned integers as the corresponding values.
        - Meanwhile, we used List<HashSet<Integer>> to store the assigned integers for each document in a list. We also d the documents names in an array in the same sequence as that list.
    - **Procedure**
        - Read all files in the given directory
        - For each file {
            - Remove punctuation symbols and the word "the".
            - Store every word to all terms HashMap<String, Ineteger> and the corresponding HashSet<Integer> in the list.
        - }

- Your procedure to assign an integer to each term.
    - For each term {
        - If the term is not in the HashMap<String, Integer>
            - Assign the size of the HashMap to that term.
            - map.put(term, map.size())
    - }

- The permutations used, and the process used to generate random permutations.
    - We used the random hash functions ($ax + b\%p$) for the permutation functions, where p > the size of all terms. For every function, a and b are randomly chosen in the range of $\{1,2,3,\ldots,p\}$.

3.2 MinHashAccuracy

- Report the number of pairs for which approximate and exact similarities differ by more than for each combination. What can you conclude from these numbers?

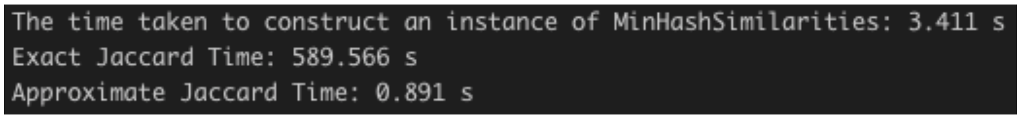


```
numPermutation: 400 epsilon: 0.04 count: 1467
numPermutation: 400 epsilon: 0.07 count: 2
numPermutation: 400 epsilon: 0.09 count: 0
numPermutation: 600 epsilon: 0.04 count: 374
numPermutation: 600 epsilon: 0.07 count: 0
numPermutation: 600 epsilon: 0.09 count: 0
numPermutation: 800 epsilon: 0.04 count: 179
numPermutation: 800 epsilon: 0.07 count: 0
numPermutation: 800 epsilon: 0.09 count: 0
```

- When the number of permutations is 400, almost similarity differences are between 0.04 and 0.07. And for 600 and 800, almost of them are under 0.04. As we increase the range of error, there are more differences between exact and approximate similarities.
- By those outputs, we can conclude that the differences count between exact and approximate similarities decreases as the number of permutations increases when the error range is fixed.

3.3 MinHashTime

- Use 600 permutations on files from space.zip. Report the total run time to calculate exact Jaccard similarities and approximate Jaccard similarities (between all possible pairs).



```
The time taken to construct an instance of MinHashSimilarities: 3.411 s
Exact Jaccard Time: 589.566 s
Approximate Jaccard Time: 0.891 s
```

  - Exact Jaccard similarities need to compare all the terms in the term-document matrix between two documents.
  - Approximate Jaccard similarities need to compare all min[Π(Di)] in the MinHash Matrix between two documents.
  - Since the size of term-document matrix is M * N, where M is the size of all terms and N is the size of all documents. And the size of MinHash Matrix is k * N, where is k is the number of permutations.
  - The time complexity of Approximate Jaccard similarities is O(k).
  - The time complexity of Exact Jaccard similarities is O(M).
  - Since O(M) > O(k), then Approximate Jaccard similarities is much faster.

4. LSH

- For the table size of each band, we choose a prime number larger than 8 * N.
- We use List<HashMap<Integer, HashSet<String>>> for the hash tables. The hashed value of each band will store in the HashMap as a key with document name as its corresponding value.
- For the hash function h, we define:
  - h(<x1, x2, … xr>) = "#x1#x2#… #xr".hashCode() mod the table size.
- For nearDuplicatesOf(String docName), we check all HashSet<String> in the HashMap. If the set contains the given document's name, the rest of the names in the set is similar to the given document.

4.1 NearDuplicates

- To find a proper number of bands, we calculate it by increments of 1 for each iteration.

- Finally, run nearDuplicateDetector on the files from F17PA2.zip (with at least two choices of s). Run the program on at least 10 different inputs For each input: List all the files that are returned as near duplicates in your report.

Num of Bands	Input file	numPermutation	Threshold	Similar docs
26	hockey111.txt	400	0.8	hockey111.txt.copy1 hockey111.txt.copy2 hockey111.txt.copy7 hockey111.txt.copy3 hockey111.txt.copy4 hockey111.txt.copy5 hockey111.txt.copy6
35	hockey111.txt	600	0.8	hockey111.txt.copy1 hockey111.txt.copy2 hockey111.txt.copy7 baseball41.txt.copy4 hockey111.txt.copy3 hockey111.txt.copy4 hockey111.txt.copy5 hockey111.txt.copy6
20	hockey111.txt	600	0.9	hockey111.txt.copy1 hockey111.txt.copy2 hockey111.txt.copy7 hockey111.txt.copy3 hockey111.txt.copy4 hockey111.txt.copy5 hockey111.txt.copy6
26	hockey661.txt	400	0.8	hockey661.txt.copy6 hockey661.txt.copy5 hockey661.txt.copy4 hockey661.txt.copy3 hockey661.txt.copy7

				hockey661.txt.copy2 hockey661.txt.copy1
15	hockey661.txt	400	0.9	hockey661.txt.copy6 hockey661.txt.copy5 hockey661.txt.copy4 hockey661.txt.copy3 hockey661.txt.copy7 hockey661.txt.copy2 hockey661.txt.copy1
26	space-101.txt	400	0.8	space-101.txt.copy6 space-101.txt.copy7 baseball737.txt.copy1 space-101.txt.copy2 space-101.txt.copy3 space-101.txt.copy4 space-101.txt.copy5 space-101.txt.copy1
35	space-101.txt	600	0.8	space-101.txt.copy6 space-101.txt.copy7 space-101.txt.copy2 space-101.txt.copy3 space-101.txt.copy4 space-573.txt.copy6 space-101.txt.copy5 space-101.txt.copy1
15	space-101.txt	400	0.9	space-101.txt.copy6 space-101.txt.copy7 space-101.txt.copy2 space-101.txt.copy3 space-101.txt.copy4 space-101.txt.copy5 space-101.txt.copy1
35	baseball1.txt	600	0.8	hockey558.txt.copy5 baseball1.txt.copy4 baseball1.txt.copy5 baseball1.txt.copy6 baseball1.txt.copy7 baseball1.txt.copy1 baseball1.txt.copy2 baseball1.txt.copy3
15	baseball1.txt	400	0.9	baseball1.txt.copy4 baseball1.txt.copy5 baseball1.txt.copy7 baseball1.txt.copy1 baseball1.txt.copy2 baseball1.txt.copy3