**Document Similarity using Shingling, Min-Hashing & Locally Sensitive Hashing**

In this project the technique of Shingling, Jaccard Similarity, Min Hashing, and LSH is used to find the document similarity.

**Code Execution**

**python <lastname>\_<firstname>\_DocSimilarity.py <folder\_path\_containing\_docs> <k> <type\_of\_shingles> <no\_of\_hashes\_for\_minhashing> <threshold\_value>**

The input parameters supplied are as follows:

• **folder\_path\_containing\_docs**: The path to the folder containing the text files.

• **k**: Value of k used while making k-shingles.

• **type\_of\_shingles:** The type of shingles you have to consider. The two possible values are **char/word.**

* If the input parameter is set as **char**, you have to construct **k-shingles** based on characters.
* If the input parameter is set as **word**, you have to construct **k-shingles** based on words.

• **no\_of\_hashes\_for\_minhashing:** No of hash functions to be used for Min-hashing.

• **threshold\_value:** The threshold value **s** used in LSH which defines how similar the documents have to be for them to be considered as similar pair.

**Sample Statements:**

**• python nayak\_nandan\_DocSimilarity.py Docs 5 char 120 0.35**

And compare your output against **sample\_char\_output.txt**

**• python nayak\_nandan\_DocSimilarity.py Docs 2 word 30 0.35**

And compare your output against **sample\_word\_output.txt**

**Algorithm Implementation**

**Task 1: Shingling**

The objective is to construct **k-shingles** for all documents. The shingles are stored only once. Duplicates are ignored. The value of **k** will be provided as an input parameter. Also the type of shingles that areconstructed will be specified. It can be either **char** or **word.**

• For input parameters **k = 5** and **type\_of\_shingles = char**, the shingles for text **“How are you”** will be:

['How a', 'ow ar', 'w are', ' are ', 'are y', 're yo', 'e you']

• For input parameters **k = 2** and **type\_of\_shingles = word**, the shingles for text **“How are you”** will be:

['Howare', 'areyou']

**Output:**

The aim is to output the **No of shingles** for each document and the **Jaccard similarity** between each possible pair of documents.

So if there are 4 different documents the number of possible pairs will be 6.

JaccardSimilarity(D1, D2) = (Shingles(D1) Ω Shingles(D2) )/( Shingles(D1) Ʊ Shingles(D2))

The **sample\_char\_output.txt** has the output format displayed.

**Task 2: Min Hashing**

Min-hashing converts the shingles to short signatures. The number of hash functions to be used will be specified as an input parameter.

Hash function to be used: **hα(x)= αx+1 mod <no\_of\_unique\_shingles\_in\_the\_corpus>.**

• Here **α = 1, 2, 3……(no\_of\_hashes\_for\_minhashing)**

• Here **x** represents the index of each k-shingle in an alphabetically sorted list having all shingles. Note that the index starts form 0.

So if the input parameter **no\_of\_hashes\_for\_minhashing = 3,** the hash functions to be used are:

• **x+1 mod <no\_of\_unique\_shingles\_in\_the\_corpus>**

• **2x+1 mod <no\_of\_unique\_shingles\_in\_the\_corpus>**

• **3x+1 mod <no\_of\_unique\_shingles\_in\_the\_corpus>**

Now using the min-hashes, the Jaccard Similarity between each possible pairs of documents is computed using the below formula.

Jaccard\_Similarity(A,B) = 1/h \* Εht=1 { 1 if Ai = Bi0 if Ai ≠Bi

here h is the **<no\_of\_min-hashes>**, A and B are the min-hash signature for Documents D1 and D2 respectively.

**Output:**

The **Min-Hash signature** for each document and the **Jaccard similarity** (obtained by the above formula) between each possible pair of documents is dipalyed. So if there are 4 different documents the number of possible pairs will be 6.

The **sample\_char\_output.txt** has the output format displayed.

**Task 3: Locality Sensitive hashing**

LSH is used to identify the candidate pairs which should actually be checked for similarity. We tune the values of **b (no of bands)** and **r (no of rows per band)** so that we have no false negatives and try to reduce the number of false positives.

• Here we will use given the value of threshold **s**, and obtain the optimal value of **b** and **r**. As we know that

**b\*r = n,** where **n** is the length of minhash signature and the approximate value of threshold **s = (1/b)^(1/r)**, we can find optimal value of b and r

• As the values for **b** and **r** are computed, the min-hash signature matrix (obtained in previous task) is split into **b bands** having **r rows** each.

• Now each band is hashed into portion of each column to a hash table with **m** buckets (Keep the value of **m** as large as possible).

• Each band is sorted into alphabetical order and considered as a tuple and inputted into the hash table to get the values. This values are used to put them into bucket.

• Now the candidate pairs of documents are those that hash to the same bucket for **>= 1** band.

**Output:**

• All possible set of candidate pairs which are obtained after performing LSH are displayed.

• Please refer to the **sample\_char\_output.txt** to find out the output .