A Project Report

On

Locality Sensitive Hashing

BY

Under the supervision of

Dr. Aruna Malapati

SUBMITTED IN PARTIAL FULLFILLMENT OF THE REQUIREMENTS OF

CS F469: Information Retrieval

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI (RAJASTHAN)

HYDERABAD CAMPUS

(March 2023)

Group Details

Name	ID no.
Aaditya Mahesh Rathi	2020A7PS2191H
Sankalp Kulkarni	2020A7PS1097H
Akshat Oke	2020A7PS0284H
Rishi Poddar	2020A7PS1195H

ACKNOWLEDGEMENTS

We are grateful to BITS Pilani for giving us this insightful opportunity of working on such an interesting course assignment. We would also like to thank the CSIS department for allowing us to work on a problem in the domain of Information Retrieval.

Finally, we would like to express my profound gratitude to Dr. Aruna Malapati for allowing us to work under her supervision and clearing our doubts throughout the duration of the course. This project would have been impossible without her help.

Table of Contents

Contents

1. Pipeline

1.1 Dataset

We used 20newsgroup dataset. It has around 18k new articles divided into topics. We used subset of this dataset for our experimentation.

1.2 K- Shingles

First we experimented by forming character-wise shingles(k=9), but we were getting a lot of false positives .We decided to have word-wise trigram(k=3) shingles. Based on literature survey and experimentation we decided to take the value of k as 3.

1.3 Pre-processing

During normalization special characters were removed and replaced with whitespaces. This choice was done because dataset had many tokens separated by special characters. The text was then converted into lowercase.

1.4 Hashing

We used the min-hashing technique. Hash functions of the form (a*x + b) % c were generated to get true permutations. This was ensured as the pairs (a, c) and (b, c) were taken as co-primes. These were chosen randomly so the results varied over multiple runs of the program.

1.5 Time Analysis

For 1000 documents, it took 32 secs to create signature matrix. Time to find similarity was 0.1-0.5 seconds depending on hash functions and rows per band. But it took almost 3 seconds to find similarity using Jaccard distance and single-document matrix. This indicates the efficiency of LSH as number of documents increase.

2. Analysis

Query	Jaccard Similarity	50 Hash Functions	100 Hash Functions	200 Hash Functions
1	96.81	94	91	82
2	92.06	90	83	69
3	51.74	16	10	13
4	84.22	48	45	35.5
5	91.49	66	68	62.5

Table 1: Average Similarity of Query Docs for Different bands

Query	Jaccard Similarity	50 Hash Functions	100 Hash Functions	200 Hash Functions
1	96.81	0.2	0.2	0.4
2	92.06	0.2	0.6	0.6
3	51.74	0.8	1	1
4	84.22	0.8	1	1
5	91.49	0.6	0.6	0.8

Table 2: False Negatives for different number of hash Functions (Threshold = 80%)

Probability of collision is low for documents with less similarity and thus it was difficult to obtain similar documents with low similarity.

Query	Jaccard Similarity	50 Hash Functions	100 Hash Functions	200 Hash Functions
1	96.81	0	0	0
2	92.06	0	0.2	0
3	51.74	0.8	1	1
4	84.22	0.4	0.6	0.6
5	91.49	0	0.2	0.2

Table 3: False Negatives for different number of hash Functions (Threshold = 50%)

No false positives were obtained in any case.

Query	Jaccard Similarity	5 bands per row	10 bands per row
1	96.81	82	62
2	92.06	69	42
3	51.74	13	6
4	84.22	35.5	24
5	91.49	62.5	39

Table 4: Comparison of avg similarity for different number of rows per band for 200 Hash Functions

Results were more accurate for less number of rows per band. This clear from formula given below that probability of collision is high for less number of rows per band.

For r = 5, b = 40: P(collision) = 0.9999

For r = 10, b = 20: P(collision) = 0.89

$$P(colision) = 1 - (1 - s^r)^b$$