NTNU
Kunnskap for en bedre verden

Department of Computer Science

TDT4306 - Big Data Architecture

Project 2

Authors:
Aaryan Neupane
Anne Torgersen

Date: 13.03.2024

Table of Contents

1 Part 1

1.1 Extracting k-shingles

Extracted all the word k-shingles with k = 1 from each sentence. In this case we are ignoring all (definite or indefinite) articles, auxiliary verbs, prepositions and conjunctions.

1: [‘big’, ‘data’, ‘platform’, ‘students’, ‘blackboard’]
2: [‘questions’, ‘minhash’, ‘project’, ‘ntnu’, ‘students’, ‘piazza’]
3: [‘ntnu’, ‘big’, ‘data’, ‘platform’, ‘blackboard’, ‘piazza’]
4: [‘project, ‘data’, ‘students’, ‘blackboard’, ‘piazza’]

Figure 1: k-shingles for each sentence

1.2 Creating Unique Words Dictionary

Searched through the k-shingles for each sentence and extracted all unique words into a dictionary. This was sorted in alphabetical order.

Unique Words Dictionary: ['big', 'blackboard', 'data', 'minhash', 'ntnu', 'piazza', 'platform', 'project', 'questions', 'students']

Figure 2: Unique Words Dictionary in alphabetical order

1.3 Input Matrix of sentences

Creating a matrix (figure 3) with the four sentences as rows and the shingles as rows. The number 1 indicates the sentence containing the shingle, while 0 represents a sentence without it.

	1	**2**	**3**	**4**
big	1	0	1	0
blackboard	1	0	1	1
data	1	0	1	1
minhash	0	1	0	0
ntnu	0	1	1	0
piazza	0	1	1	1
platform	1	0	1	0
project	0	1	0	1
questions	0	1	0	0
students	1	1	0	1

Figure 3: Input Matrix of sentences

Figure 4 presents all the computed jaccard similarities between sentence pairs.

	1	2	3	4
1	1	1/10	4/7	3/7
2	1/10	1	2/10	3/8
3	4/7	2/10	1	3/8
4	3/7	3/8	3/8	1

Figure 4: Jaccard Similarities

1.4 Calculating Min-Hash Signature Matrix

H1	H2	H3
5	9	10
6	6	7
7	3	4
8	10	1
9	7	8
10	4	5
1	1	2
2	8	9
3	5	6
4	2	3

	1	2	3	4
big	1	0	1	0
blackboard	1	0	1	1
data	1	0	1	1
minhash	0	1	0	0
ntnu	0	1	1	0
piazza	0	1	1	1
platform	1	0	1	0
project	0	1	0	1
questions	0	1	0	0
students	1	1	0	1

Figure 5: Permutations to use on the input matrix

H1	1	2	1	2
H2	1	2	1	2
H3	2	1	2	3

Figure 6: Signature Matrix

From 1.3 we have that sentence pair (1,3) has the highest Jaccard similarity. This corresponds to the signature matrix in figure 6. From the signature matrix we get that the similarity between sentence pair (1,3) is 1. The similarity between sentence pair (1,2) and (1,4) are 0 and (2,4) is 2/3. This means that sentence 1 and 3 are the most similar.

2 Report

2.1 K-shingles

The parameter k determines the number of words included in each shingle. When k = 1, each word forms a shingle, whereas when k = 5, every group of five consecutive words forms a shingle. We observe that k = 1 gives 65553 shingles, k = 5 gives 759983 shingles, and k = 10 gives 760532 shingles. This implies that bigger k-values results in more shingles. This makes sense because larger k-values encompass more words within each shingle, thereby increasing the number of possible combinations and, consequently, the total count of shingles.

Furthermore, when evaluating performance, we find that k = 1 exhibits the poorest results, characterized by a significant gap between candidate pairs and true pairs, along with a high rate of false positives. While the difference in performance between k = 5 and k = 10 is less pronounced, k = 5 demonstrates fewer false negatives, indicating its slightly superior performance in this regard.

2.2 Number of permutations

The permutations parameter determines the number of different hash functions to be applied. A higher number of permutations generally reduces the number of false positives in LSH by improving the ability to distinguish between similar and dissimilar documents. The hash functions become more discriminating, which means that similar documents are more likely to hash to the same buckets, increasing the chances of true positives and decreasing the chances of false positives. However, there's a trade-off with computational cost when using a higher number of permutations.

We opted to exclude the values 10 and 50 for p in testing, as utilizing these values would result in an unequal division of rows for each band due to the constraints imposed by the other default parameters.

2.3 Number of bands

The parameter 'bands' in the LSH algorithm determines the number of divisions of the signature matrix into multiple segments for comparison. With each iteration through the bands, document columns in the signature matrix are hashed into buckets. A higher number of bands results in fewer rows per band, consequently increasing the likelihood of document pairs hashing into the same bucket, thereby boosting the number of potential candidate pairs.

Our experiments, detailed in the Appendix[2.4], revealed that the default parameter value of b=20 yields the highest number of true pairs while maintaining a low rate of false negatives. This value emerged as the optimal choice from our tests.

2.4 Naive vs LSH

The naive method operates with a complexity of O(n^2), where n denotes the number of documents, which in this project is 2225. This quadratic time complexity makes it impractical for larger datasets, as the time required grows exponentially with the dataset size. The naive method's reliance on pairwise comparisons becomes prohibitive as the number of documents increases, limiting its scalability and real-world applicability.

In contrast, the LSH method offers a more efficient alternative. Typically, its complexity remains O(n) or even sub-linear. By leveraging hash functions to group similar documents into buckets, LSH reduces the number of comparisons necessary for similarity computation. This approach ensures reasonable execution times, even with substantial datasets, making it highly scalable and suitable for practical usage.

Comparing both methods, while the naive approach is conceptually straightforward, its inefficiency hampers its usefulness for large-scale document collections. On the other hand, LSH's efficiency and scalability make it the preferred choice for document similarity analysis, especially in scenarios involving sizable datasets.

In summary, the LSH method stands out for its superior running time efficiency, offering a practical solution for document similarity computation across various scales of data.

Appendix

A Results from the default parameters

```
The number of uniquek-shingles of each document for k = 5: 759983
```

```
Starting to get the pairs of documents with over  0.6 % similarity...
The total number of candidate pairs from LSH: 170
The total number of true pairs from LSH: 168
The total number of false positives from LSH: 2
```

Figure 7: Results from the default parameters

B Results from K = 1

```
The number of k-shingles of each document for k = 1: 65553
```

```
Starting to get the pairs of documents with over  0.6 % similarity...
The total number of candidate pairs from LSH: 336
The total number of true pairs from LSH: 173
The total number of false positives from LSH: 163
```

Figure 8: Results for $k = 1$

C Results from K = 10

```
The number of unique k-shingles of each document for k = 10: 760532
```

```
Starting to get the pairs of documents with over  0.6 % similarity...
The total number of candidate pairs from LSH: 166
The total number of true pairs from LSH: 159
The total number of false positives from LSH: 7
```

Figure 9: Results for $k = 10$

D Results from permutations = 20

```
Starting to get the pairs of documents with over  0.6 % similarity...
The total number of candidate pairs from LSH: 614
The total number of true pairs from LSH: 169
The total number of false positives from LSH: 445
```

Figure 10: Results from p = 20

E Results from bands = 50, 100

```
Starting to get the pairs of documents with over  0.6 % similarity...
The total number of candidate pairs from LSH: 244
The total number of true pairs from LSH: 168
The total number of false positives from LSH: 76
```

(a) $b = 50$

```
Starting to get the pairs of documents with over  0.6 % similarity...
The total number of candidate pairs from LSH: 2075
The total number of true pairs from LSH: 163
The total number of false positives from LSH: 1912
```

(b) $b = 100$

Figure 11: Results for $b = 50$ and $b = 100$