ID2222 Homework 1 Report

Finding Similar Items: Textually Similar Documents

Group 66: Tingyu Lei, Yiyao Zhang

Solution

We utilize Python and its built-in functions for this homework. We have implemented shingling, min-hashing and locality-sensitive hashing (LSH) and compare functions to compare the similarity and signature similarity. We also have a main file invoking all three methods to demonstrate our results.

Dataset

[Twenty Newsgroups - UCI Machine Learning Repository]

Core Classes

Shingling class constructs k-shingles from a document, computes hash values for each unique shingle, and represents the document as an ordered set of hashed k-shingles.

```
class Shingling:

    def __init__(self, k: int = 10):
      ..
    def create_shingles(self, doc: str) -> Set[int]:
      ..
```

create_shingles creates k-shingles from a document and returns a set of hashed shingles.

CompareSets class computes the Jaccard similarity of two sets of integers (sets of hashed shingles).

```
class CompareSets:
    @staticmethod
    def jaccard_similarity(set1: Set[int], set2: Set[int]) -> float:
      ..
```

CompareSignatures class that estimates the similarity of two integer vectors (minhash signatures) as a fraction of components in which they agree. This approximates the Jaccard similarity of the original sets.

```
class CompareSignatures:
    @staticmethod
    def signature_similarity(sig1: List[int], sig2: List[int]) -> float:
      ..
```

MinHashing class builds a minHash signature (vector) of a given length n from a given set of integers (hashed shingles).

```
class MinHashing:
    def __init__(self, num_permutations: int = 100, seed: int = 42):
       ...
    def compute_signature(self, shingles: Set[int]) -> List[int]:
      ..
```

How to run

1. Extract the homework folder.
2. Install Python3
3. You can modify the following parameters in `main.py`:
    - `k`: Shingle length (default: 10)
    - `num_permutations`: Number of hash functions for minhashing (default: 100)
    - `similarity_threshold`: Threshold for considering documents similar (default: 0.8)
    - `num_docs`: Number of documents to process (default: 1000)
    - `num_bands`: Number of bands for LSH (default: 10)
    - `num_rows_per_band`: Number of rows per band for LSH (default: 10)
4. Run `python main.py` in the current folder.

Results

With the default configuration, the results are as follows:

```
1 python .\main.py
Config:
  Shingle length: 10
  Number of permutations: 100
  Similarity threshold: 0.8
  Number of docs: 1000
  LSH bands: 10, rows per band: 10

Loading docs...

Shingling and Jaccard Similarity
Execution time: 69.9654 seconds
Found 10 pairs:
  Documents 0 and 900: similarity = 0.9299
  Documents 205 and 901: similarity = 0.9184
  Documents 389 and 390: similarity = 0.5999
  Documents 579 and 646: similarity = 0.5557
  Documents 367 and 372: similarity = 0.5477
  Documents 212 and 266: similarity = 0.5473
  Documents 252 and 603: similarity = 0.5446
  Documents 266 and 536: similarity = 0.5413
  Documents 458 and 462: similarity = 0.5346
  Documents 368 and 375: similarity = 0.5167

Found 2 similar pairs:
  Documents 0 and 900: similarity = 0.9299
  Documents 205 and 901: similarity = 0.9184

MinHash Signatures
Execution time: 34.0948 seconds
Found 10 pairs:
  Documents 0 and 900: similarity = 0.9400
  Documents 205 and 901: similarity = 0.9300
  Documents 579 and 646: similarity = 0.6200
  Documents 49 and 171: similarity = 0.5800
  Documents 252 and 603: similarity = 0.5800
  Documents 458 and 462: similarity = 0.5700
  Documents 212 and 266: similarity = 0.5600
  Documents 579 and 630: similarity = 0.5600
  Documents 389 and 390: similarity = 0.5600
  Documents 565 and 842: similarity = 0.5600

Found 2 similar pairs:
  Documents 0 and 900: similarity = 0.9400
  Documents 205 and 901: similarity = 0.9300

LSH
Execution time: 32.5905 seconds
Found 2 pairs:
  Documents 0 and 900: similarity = 0.9400
  Documents 205 and 901: similarity = 0.9300

Found 2 similar pairs:
  Documents 0 and 900: similarity = 0.9400
  Documents 205 and 901: similarity = 0.9300

Summary
Shingling time: 69.9654 seconds
MinHash time: 34.0948 seconds
LSH time: 32.5905 seconds

Speedup (MinHash vs Shingling): 2.05x
Speedup (LSH vs Shingling): 2.15x

Pair comparison:
  Shingling pairs: 2
  MinHash pairs: 2
  LSH pairs: 2
  MinHash matches Shingling: True
  LSH candidate pairs: 2 (may include false positives)
```

If we only use 100 documents to process, the results are as follows:

```
1 python .\main.py
Config:
  Shingle length: 10
  Number of permutations: 100
  Similarity threshold: 0.8
  Number of docs: 100
  LSH bands: 10, rows per band: 10

Loading docs...

Shingling and Jaccard Similarity
Execution time: 0.5899 seconds
Found 10 pairs:
  Documents 10 and 64: similarity = 0.4352
  Documents 8 and 39: similarity = 0.4053
  Documents 80 and 87: similarity = 0.3917
  Documents 32 and 33: similarity = 0.3742
  Documents 26 and 62: similarity = 0.3680
  Documents 69 and 93: similarity = 0.3674
  Documents 9 and 10: similarity = 0.3553
  Documents 58 and 75: similarity = 0.3522
  Documents 9 and 63: similarity = 0.3412
  Documents 7 and 69: similarity = 0.3384

Found 0 similar pairs:

MinHash Signatures
Execution time: 3.3157 seconds
Found 10 pairs:
  Documents 10 and 64: similarity = 0.4300
  Documents 26 and 62: similarity = 0.4200
  Documents 9 and 10: similarity = 0.4200
  Documents 8 and 39: similarity = 0.4100
  Documents 80 and 87: similarity = 0.4100
  Documents 58 and 75: similarity = 0.4000
  Documents 69 and 93: similarity = 0.3800
  Documents 7 and 69: similarity = 0.3500
  Documents 32 and 33: similarity = 0.3500
  Documents 73 and 74: similarity = 0.3200

Found 0 similar pairs:

LSH
Execution time: 3.3266 seconds
Found 0 pairs:

Found 0 similar pairs:

Summary
Shingling time: 0.5899 seconds
MinHash time: 3.3157 seconds
LSH time: 3.3266 seconds

Speedup (MinHash vs Shingling): 0.18x
Speedup (LSH vs Shingling): 0.18x

Pair comparison:
  Shingling pairs: 0
  MinHash pairs: 0
  LSH pairs: 0
  MinHash matches Shingling: True
  LSH candidate pairs: 0 (may include false positives)
```

It can be observed that shingling exhibits a certain speed advantage in small-scale datasets, but when the dataset becomes very large, LSH proves more efficient.

We can also see that the similarity threshold of 0.8 is quite high and may not find many similar pairs in diverse document collections and LSH is most beneficial for

large document collections where brute force shingling comparison becomes expensive.