WSM Project 2: Building IR systems based on the Pyserini Project

WSM 1131

Toolkits

Lucene: https://github.com/apache/lucene

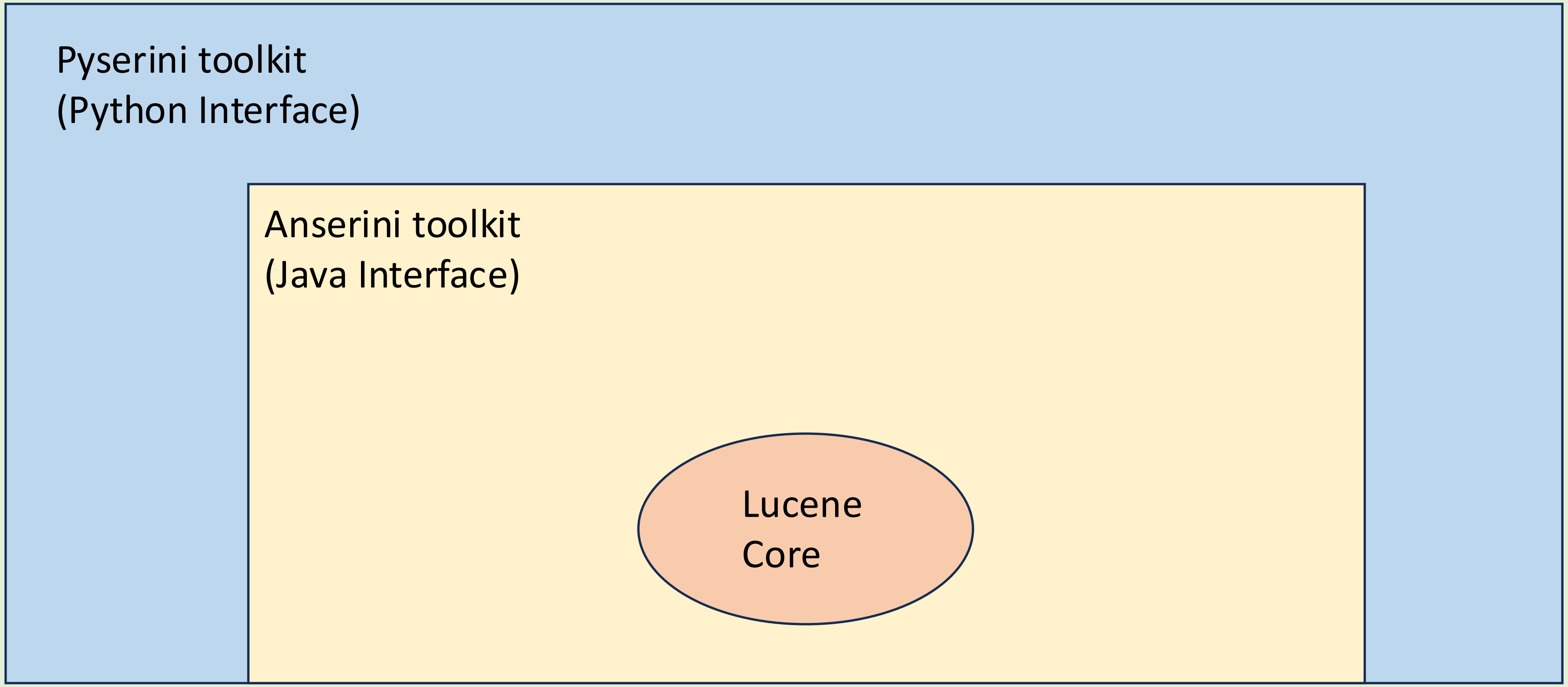
Anserini: https://github.com/castorini/anserini

Pyserini: https://github.com/castorini/pyserini

Pyserini documentations:

https://github.com/castorini/pyserini/blob/master/docs/usage-index.md

Relation



Pyserini toolkit
(Python Interface)

Anserini toolkit
(Java Interface)

Lucene
Core

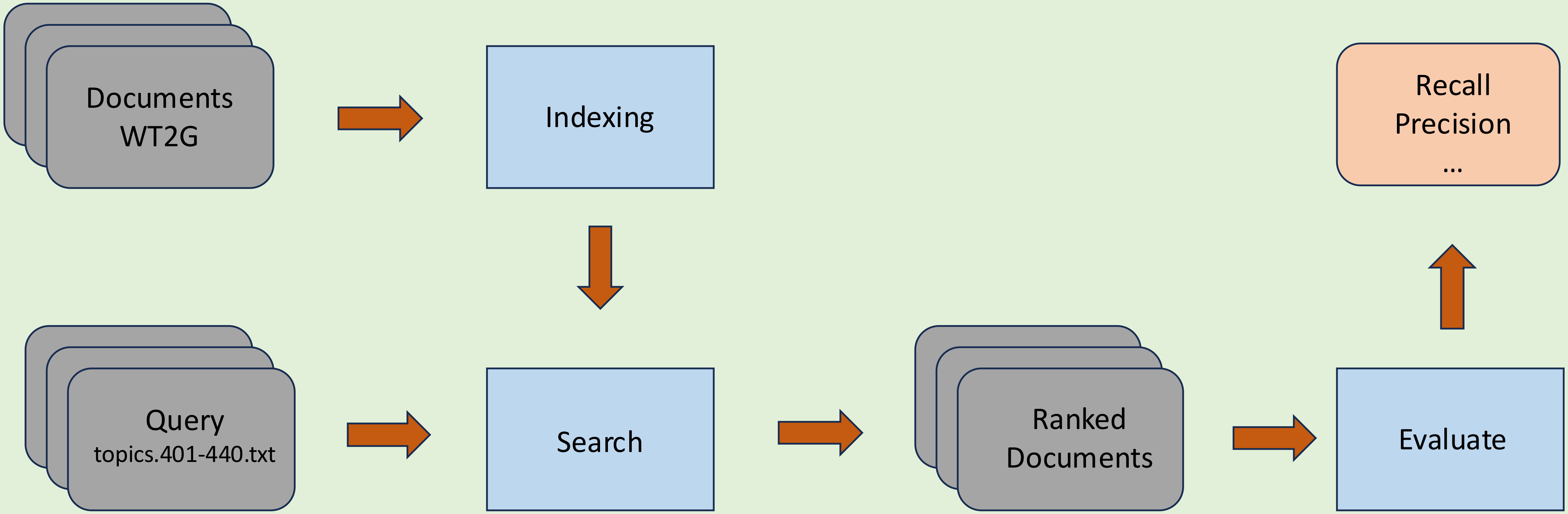
Pyserini Installation

For windows, we recommend to use wsl:

https://learn.microsoft.com/zh-tw/windows/wsl/install

- OpenJDK
  - sudo apt-get update
  - sudo apt-get install default-jre
  - sudo apt-get install default-jdk
  - java --version
- pip install pyserini
  - if you encounter missing module: torch, faiss
    - pip install torch
    - pip install faiss-cpu

Task



Documents
WT2G

Indexing

Query
topics.401-440.txt

Search

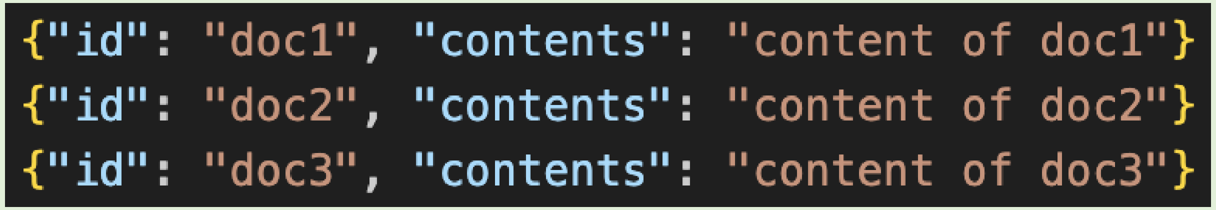
Ranked
Documents

Evaluate

Recall
Precision
…

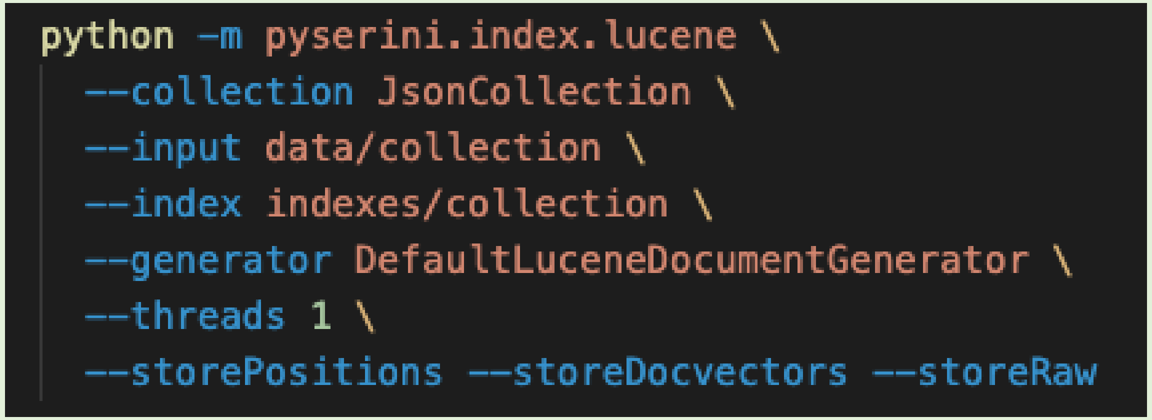
Build Index Collections

Covert the corpus into jsonl format and store as *collections.jsonl*:

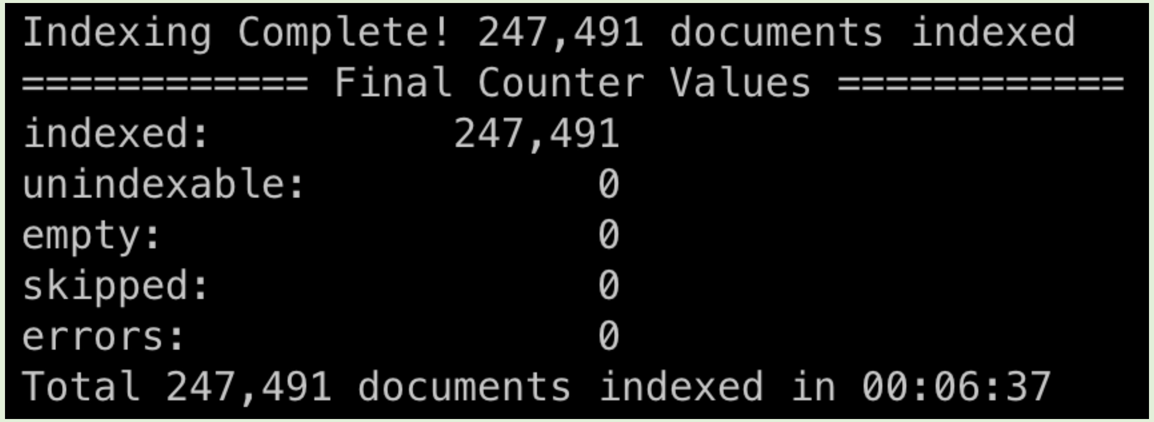


```
{"id": "doc1", "contents": "content of doc1"}
{"id": "doc2", "contents": "content of doc2"}
{"id": "doc3", "contents": "content of doc3"}
```

Put the jsonl file in *data/collection/collections.jsonl* and start building index:

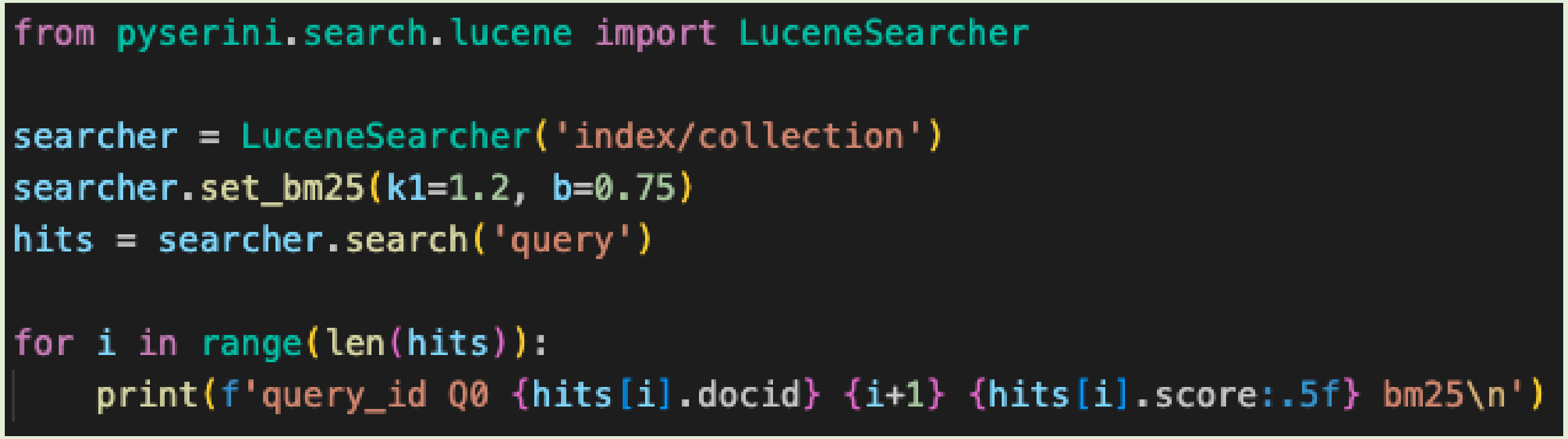


```
python -m pyserini.index.lucene \
  --collection JsonCollection \
  --input data/collection \
  --index indexes/collection \
  --generator DefaultLuceneDocumentGenerator \
  --threads 1 \
  --storePositions --storeDocvectors --storeRaw
```



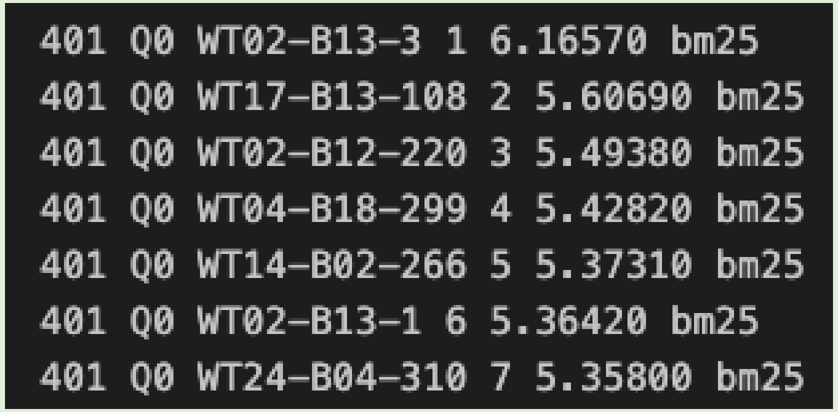
```
Indexing Complete! 247,491 documents indexed
============ Final Counter Values ============
indexed:          247,491
unindexable:            0
empty:                  0
skipped:                0
errors:                 0
Total 247,491 documents indexed in 00:06:37
```

Search Collections



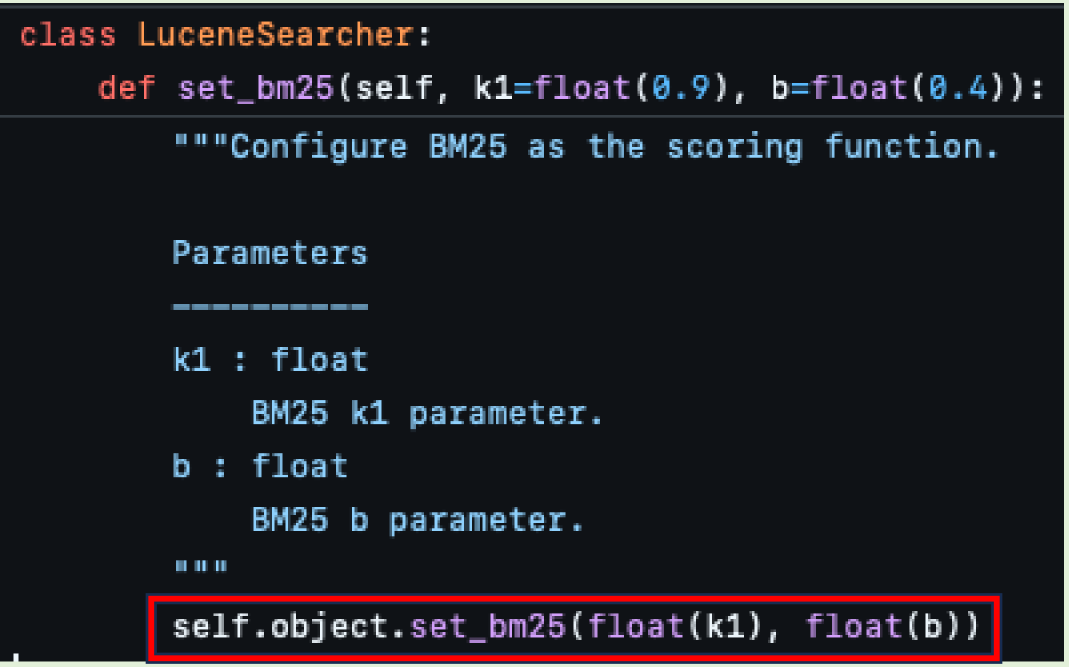
```
from pyserini.search.lucene import LuceneSearcher

searcher = LuceneSearcher('index/collection')
searcher.set_bm25(k1=1.2, b=0.75)
hits = searcher.search('query')

for i in range(len(hits)):
    print(f'query_id Q0 {hits[i].docid} {i+1} {hits[i].score:.5f} bm25\n')
```

TREC format:
query-id Q0 document-id rank score Exp

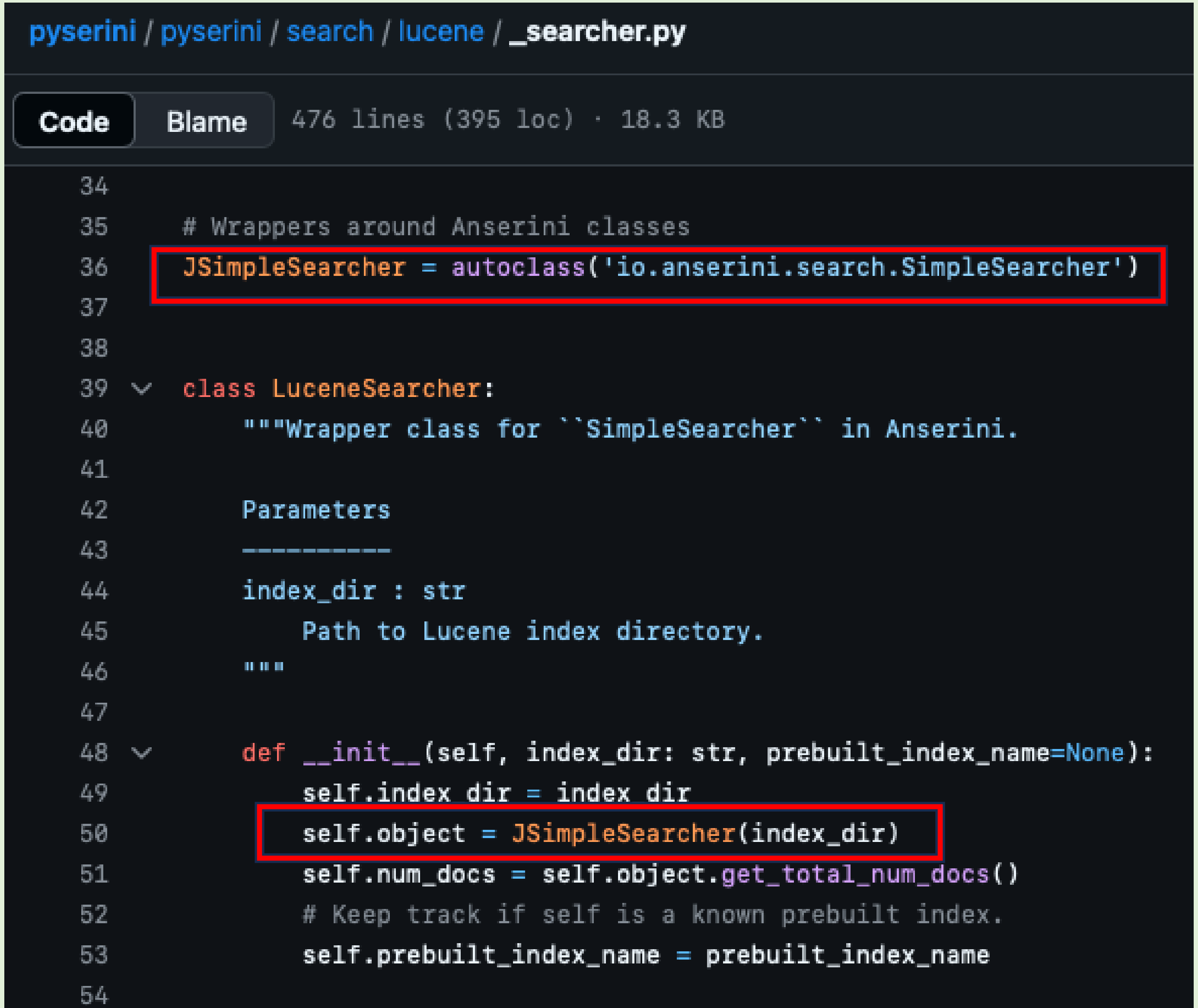


```
401 Q0 WT02-B13-3 1 6.16570 bm25
401 Q0 WT17-B13-108 2 5.60690 bm25
401 Q0 WT02-B12-220 3 5.49380 bm25
401 Q0 WT04-B18-299 4 5.42820 bm25
401 Q0 WT14-B02-266 5 5.37310 bm25
401 Q0 WT02-B13-1 6 5.36420 bm25
401 Q0 WT24-B04-310 7 5.35800 bm25
```

Search Collections - Different Ranking Function



```
class LuceneSearcher:
    def set_bm25(self, k1=float(0.9), b=float(0.4)):
        """Configure BM25 as the scoring function.

        Parameters
        ----------
        k1 : float
            BM25 k1 parameter.
        b : float
            BM25 b parameter.
        """
        self.object.set_bm25(float(k1), float(b))
```



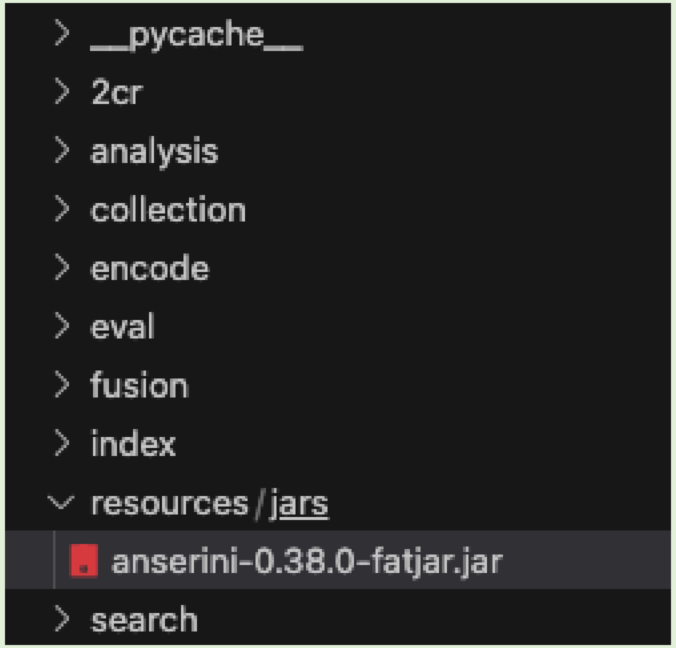
pyserini / pyserini / search / lucene / _searcher.py
Code
Blame
476 lines (395 loc) · 18.3 KB

```

# Wrappers around Anserini classes
JSimpleSearcher = autoclass('io.anserini.search.SimpleSearcher')


class LuceneSearcher:
    """Wrapper class for ``SimpleSearcher`` in Anserini.

    Parameters
    ----------
    index_dir : str
        Path to Lucene index directory.
    """

    def __init__(self, index_dir: str, prebuilt_index_name=None):
        self.index_dir = index_dir
        self.object = JSimpleSearcher(index_dir)
        self.num_docs = self.object.get_total_num_docs()
        # Keep track if self is a known prebuilt index.
        self.prebuilt_index_name = prebuilt_index_name

```

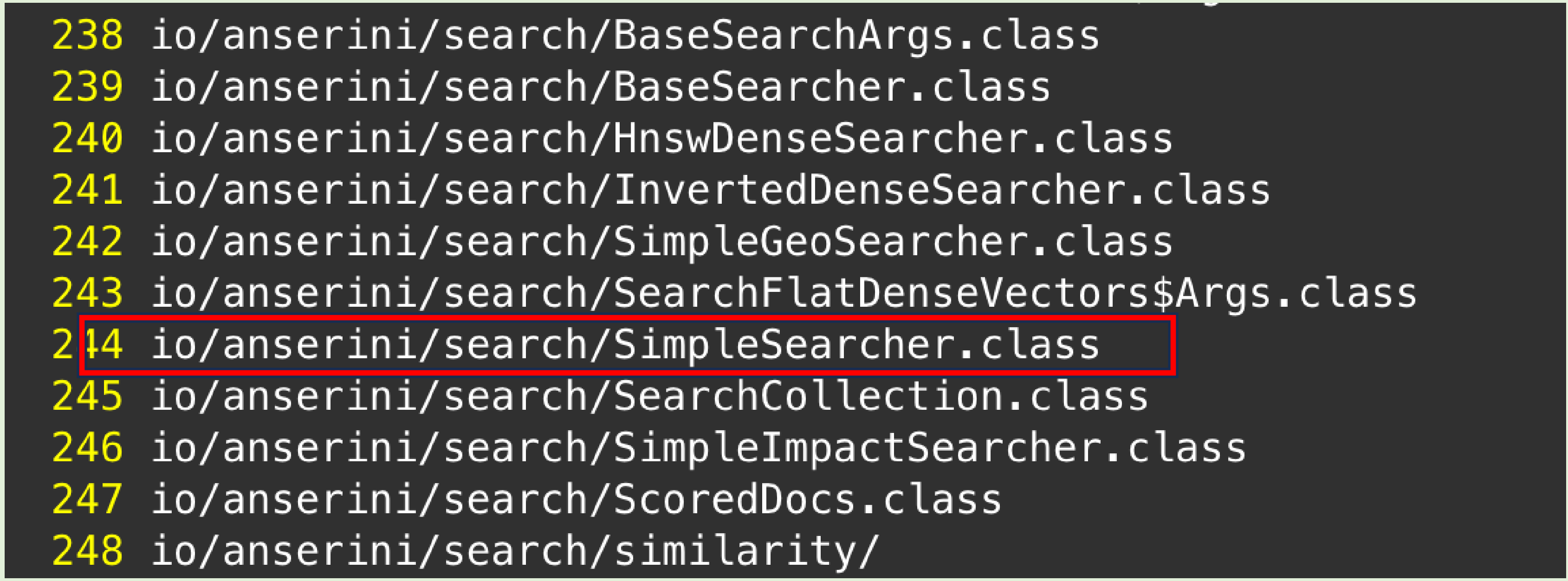
Search Collections - Different Ranking Function

How to use Lucene / Anserini JAVA Class?

https://pypi.org/project/pyjnius/



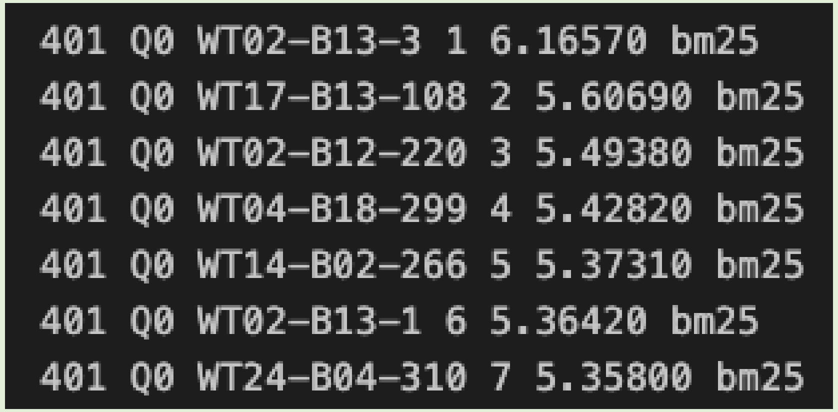
```
> __pycache__
> 2cr
> analysis
> collection
> encode
> eval
> fusion
> index
v resources / jars
    anserini-0.38.0-fatjar.jar
> search
```



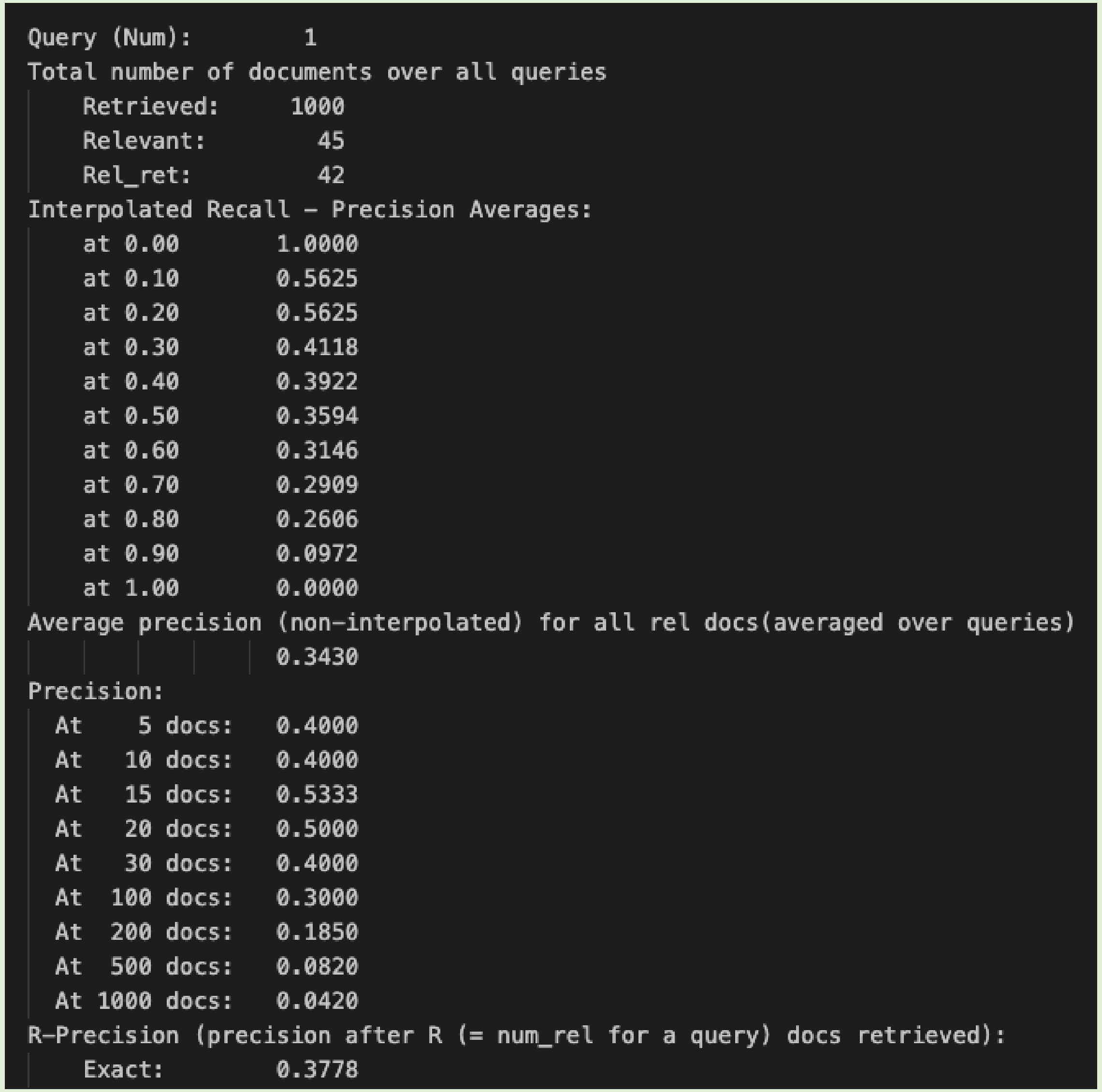
```
238 io/anserini/search/BaseSearchArgs.class
239 io/anserini/search/BaseSearcher.class
240 io/anserini/search/HnswDenseSearcher.class
241 io/anserini/search/InvertedDenseSearcher.class
242 io/anserini/search/SimpleGeoSearcher.class
243 io/anserini/search/SearchFlatDenseVectors$Args.class
244 io/anserini/search/SimpleSearcher.class
245 io/anserini/search/SearchCollection.class
246 io/anserini/search/SimpleImpactSearcher.class
247 io/anserini/search/ScoredDocs.class
248 io/anserini/search/similarity/
```

Evaluation

trec_eval.pl



```
401 Q0 WT02-B13-3 1 6.16570 bm25
401 Q0 WT17-B13-108 2 5.60690 bm25
401 Q0 WT02-B12-220 3 5.49380 bm25
401 Q0 WT04-B18-299 4 5.42820 bm25
401 Q0 WT14-B02-266 5 5.37310 bm25
401 Q0 WT02-B13-1 6 5.36420 bm25
401 Q0 WT24-B04-310 7 5.35800 bm25
```



```
Query (Num):        1
Total number of documents over all queries
    Retrieved:     1000
    Relevant:        45
    Rel_ret:         42
Interpolated Recall - Precision Averages:
    at 0.00       1.0000
    at 0.10       0.5625
    at 0.20       0.5625
    at 0.30       0.4118
    at 0.40       0.3922
    at 0.50       0.3594
    at 0.60       0.3146
    at 0.70       0.2909
    at 0.80       0.2606
    at 0.90       0.0972
    at 1.00       0.0000
Average precision (non-interpolated) for all rel docs(averaged over queries)
                  0.3430
Precision:
  At    5 docs:   0.4000
  At   10 docs:   0.4000
  At   15 docs:   0.5333
  At   20 docs:   0.5000
  At   30 docs:   0.4000
  At  100 docs:   0.3000
  At  200 docs:   0.1850
  At  500 docs:   0.0820
  At 1000 docs:   0.0420
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.3778
```

Thank You