Development and Evaluation of an IR System

Salman Khan

Irene Kahvazadeh

Edwin Jose

Outline/features

Complete and functional IR system

Two tokenization methods

Stemming vs. no stemming

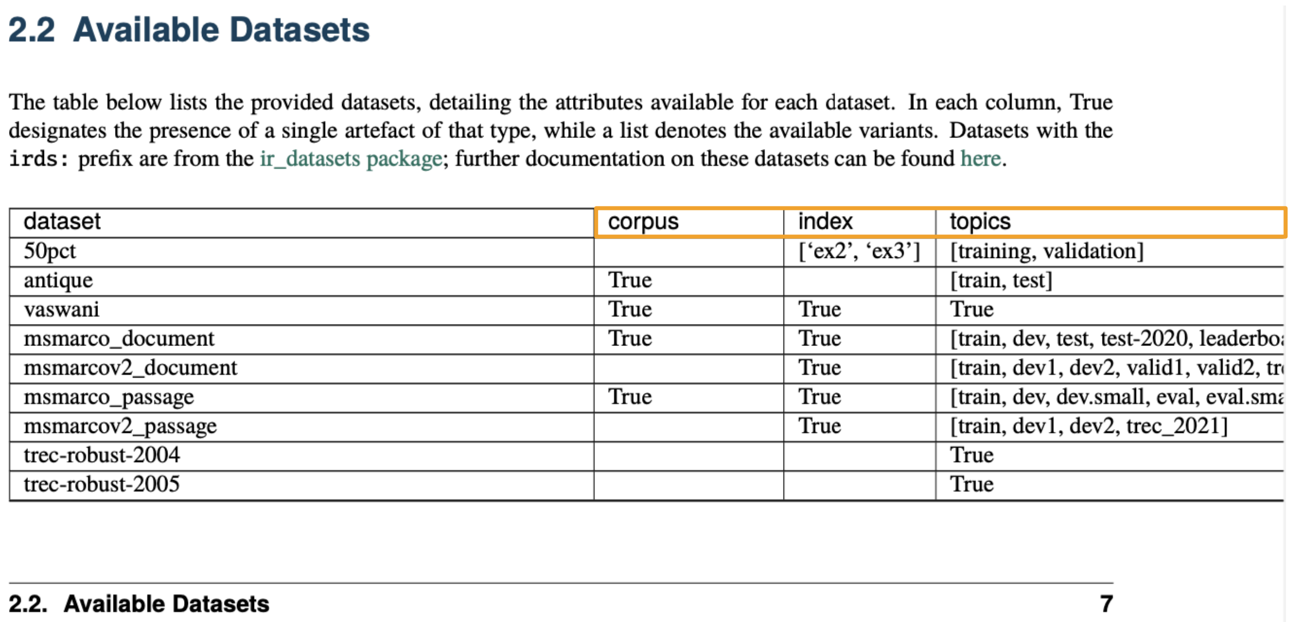
Stopword removal: yes vs. no

Comparison of ranking methods

Application of multiple performance metrics

Complete and functional IR system

```
dir_path = input("Enter the directory's path that contains the documents to search (or -1
to skip): ")

if dir_path != "-1":
    if not os.path.isdir(dir_path):
        print("Entered path does not point to a directory.")
        exit()

    docs = []

    for i, filename in enumerate(os.listdir(dir_path)):
        print("docid " + str(i) + ": " + dir_path + "/" + filename)

        docs.append(dir_path + "/" + filename)

    if os.path.exists("./temp_index"):
        shutil.rmtree("./temp_index")

    index = pt.FilesIndexer("./temp_index").index(docs)


    batch = pt.BatchRetrieve(index, wmodel="BM25")

    while True:
        query = input("\nEnter search query (or -1 to exit search): ")

        if query == "-1":
            break

        print(batch.search(query))

    if os.path.exists("./temp_index"):
        shutil.rmtree("./temp_index")
```

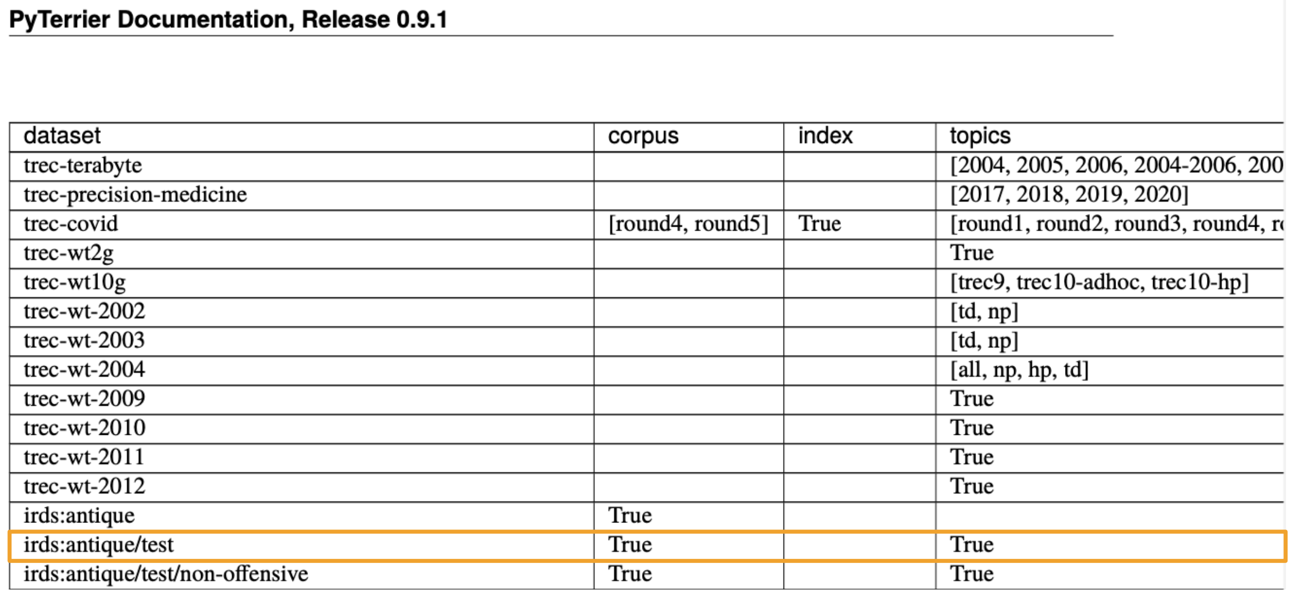


2.2 Available Datasets

The table below lists the provided datasets, detailing the attributes available for each dataset. In each column, True designates the presence of a single artefact of that type, while a list denotes the available variants. Datasets with the `irds:` prefix are from the ir_datasets package; further documentation on these datasets can be found here.

dataset	corpus	index	topics
50pct		['ex2', 'ex3']	[training, validation]
antique	True		[train, test]
vaswani	True	True	True
msmarco_document	True	True	[train, dev, test, test-2020, leaderbo
msmarcov2_document		True	[train, dev1, dev2, valid1, valid2, tr
msmarco_passage	True	True	[train, dev, dev.small, eval, eval.sma
msmarcov2_passage		True	[train, dev1, dev2, trec_2021]
trec-robust-2004			True
trec-robust-2005			True

2.2. Available Datasets 7



PyTerrier Documentation, Release 0.9.1

dataset	corpus	index	topics
trec-terabyte			[2004, 2005, 2006, 2004-2006, 200
trec-precision-medicine			[2017, 2018, 2019, 2020]
trec-covid	[round4, round5]	True	[round1, round2, round3, round4, r
trec-wt2g			True
trec-wt10g			[trec9, trec10-adhoc, trec10-hp]
trec-wt-2002			[td, np]
trec-wt-2003			[td, np]
trec-wt-2004			[all, np, hp, td]
trec-wt-2009			True
trec-wt-2010			True
trec-wt-2011			True
trec-wt-2012			True
irds:antique	True		
irds:antique/test	True		True
irds:antique/test/non-offensive	True		True

Dataset

404K docs

200 queries

Ranking methods used

TF (Term Frequency)

TF-IDF (Term Frequency–Inverse Document Frequency)

BM25 (BM = Best Matching)

Performance metrics methods used

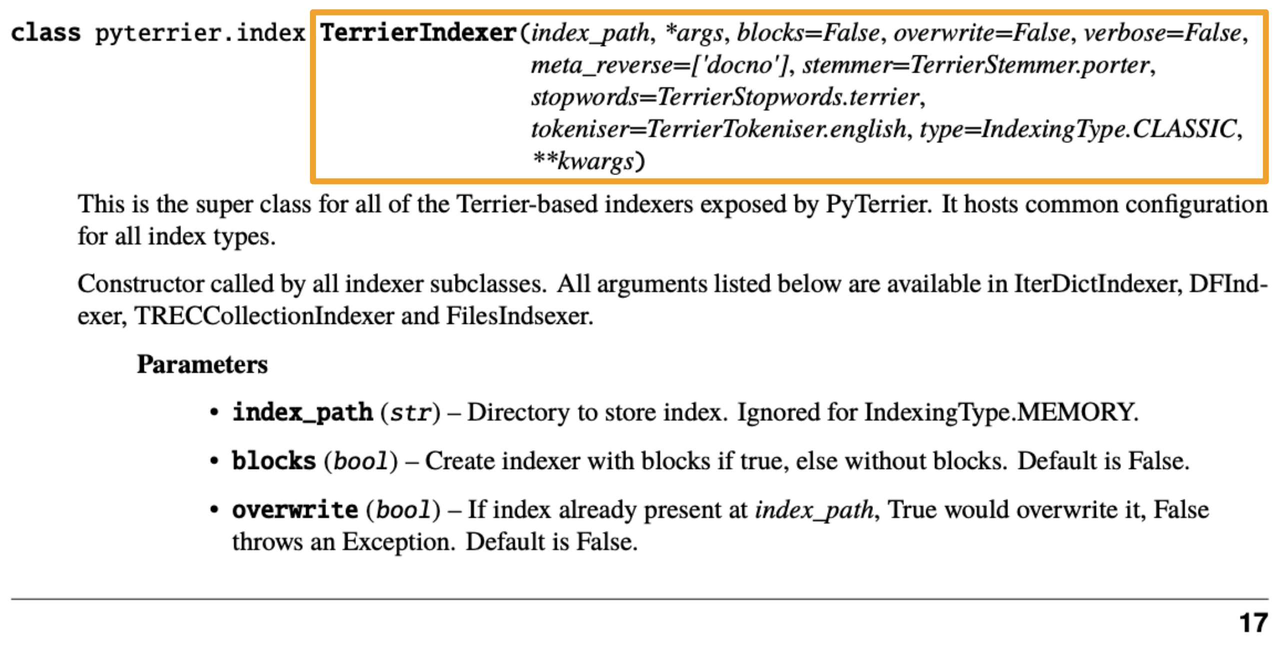
Recall@5

Recall@10

Precision@5

Precision@10

NDCG (Normalized Discounted Cumulative Gain)



```
class pyterrier.index.TerrierIndexer(index_path, *args, blocks=False, overwrite=False, verbose=False,
                                     meta_reverse=['docno'], stemmer=TerrierStemmer.porter,
                                     stopwords=TerrierStopwords.terrier,
                                     tokeniser=TerrierTokeniser.english, type=IndexingType.CLASSIC,
                                     **kwargs)
```

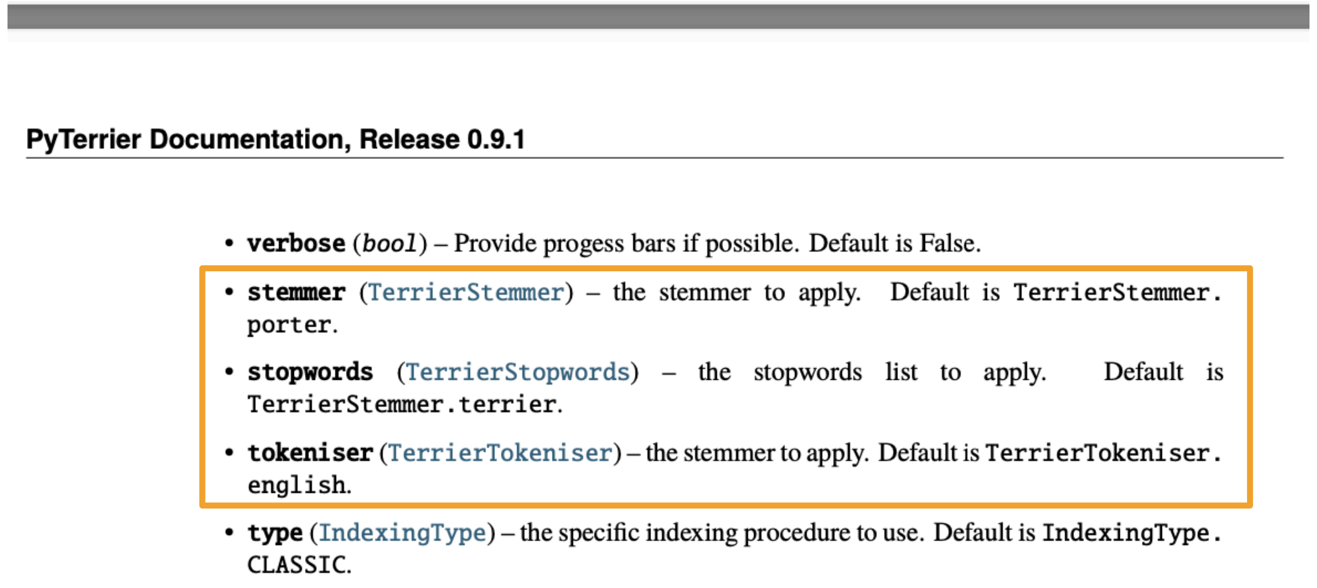
This is the super class for all of the Terrier-based indexers exposed by PyTerrier. It hosts common configuration for all index types.

Constructor called by all indexer subclasses. All arguments listed below are available in IterDictIndexer, DFIndexer, TRECCollectionIndexer and FilesIndsexer.

Parameters

- **index_path** (*str*) – Directory to store index. Ignored for IndexingType.MEMORY.
- **blocks** (*bool*) – Create indexer with blocks if true, else without blocks. Default is False.
- **overwrite** (*bool*) – If index already present at *index_path*, True would overwrite it, False throws an Exception. Default is False.

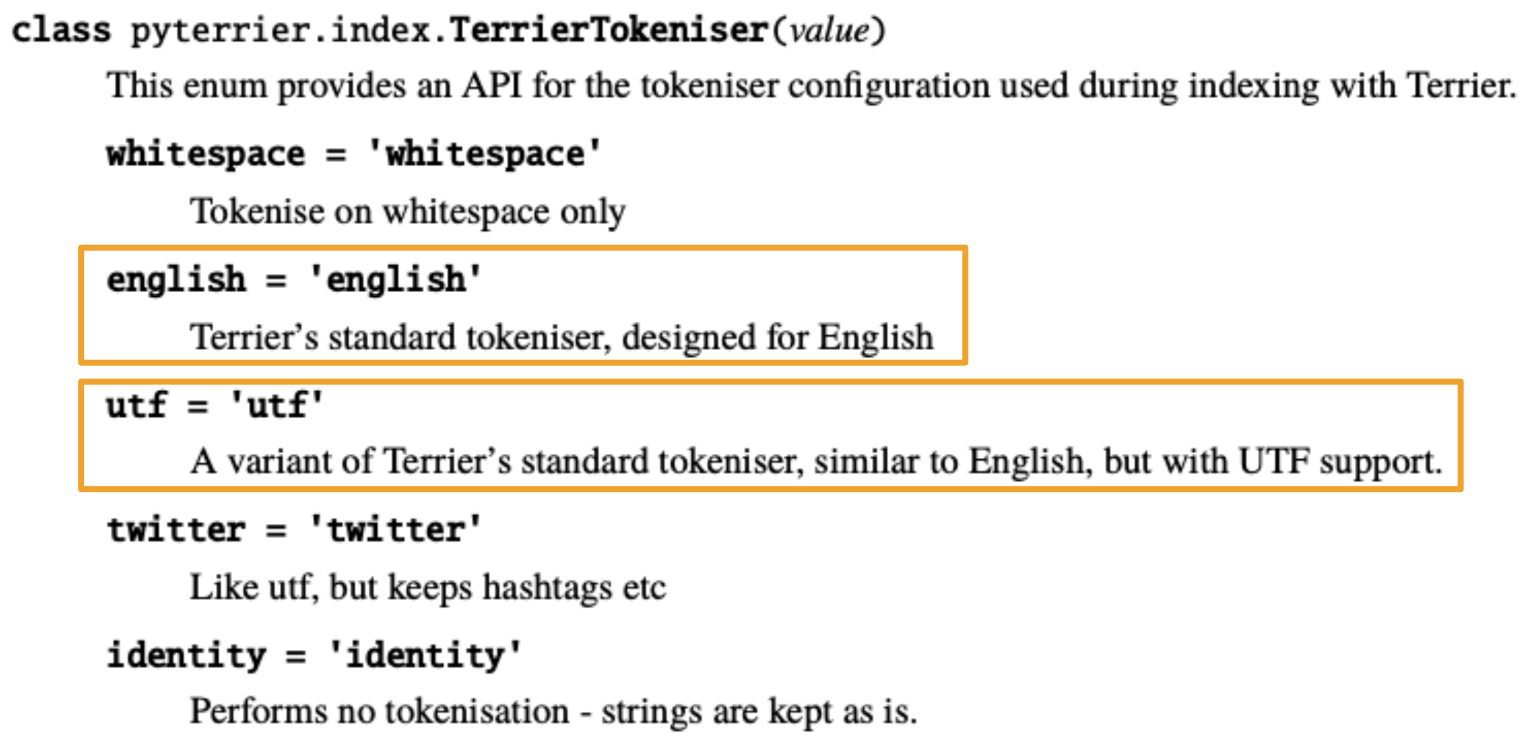
17



PyTerrier Documentation, Release 0.9.1

- **verbose** (*bool*) – Provide progess bars if possible. Default is False.
- **stemmer** (TerrierStemmer) – the stemmer to apply. Default is `TerrierStemmer.porter`.
- **stopwords** (TerrierStopwords) – the stopwords list to apply. Default is `TerrierStemmer.terrier`.
- **tokeniser** (TerrierTokeniser) – the stemmer to apply. Default is `TerrierTokeniser.english`.
- **type** (IndexingType) – the specific indexing procedure to use. Default is `IndexingType.CLASSIC`.

Indexer



```
class pyterrier.index.TerrierTokeniser(value)
```

This enum provides an API for the tokeniser configuration used during indexing with Terrier.

`whitespace = 'whitespace'`

Tokenise on whitespace only

`english = 'english'`

Terrier's standard tokeniser, designed for English

`utf = 'utf'`

A variant of Terrier's standard tokeniser, similar to English, but with UTF support.

`twitter = 'twitter'`

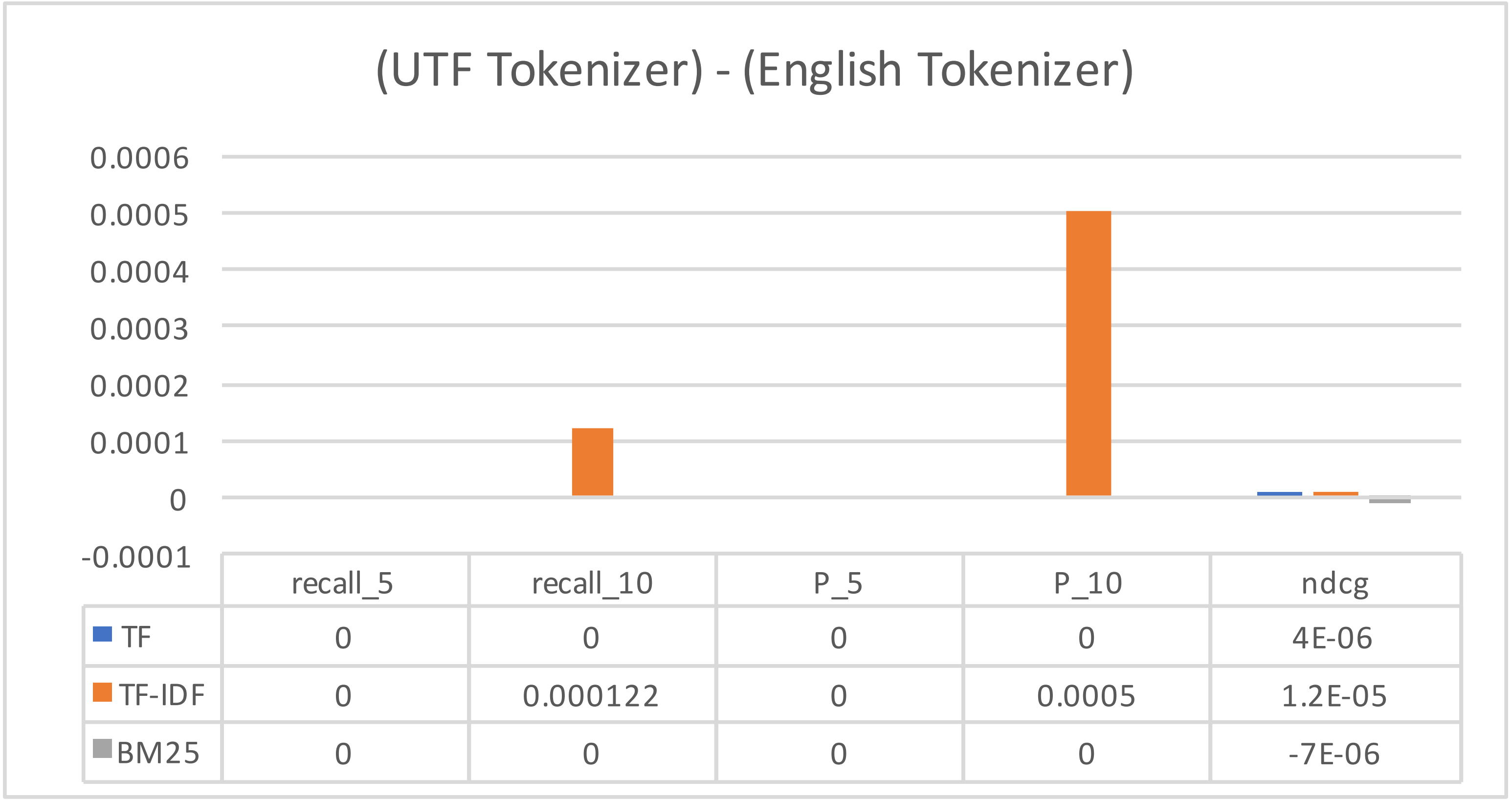
Like utf, but keeps hashtags etc

`identity = 'identity'`

Performs no tokenisation - strings are kept as is.

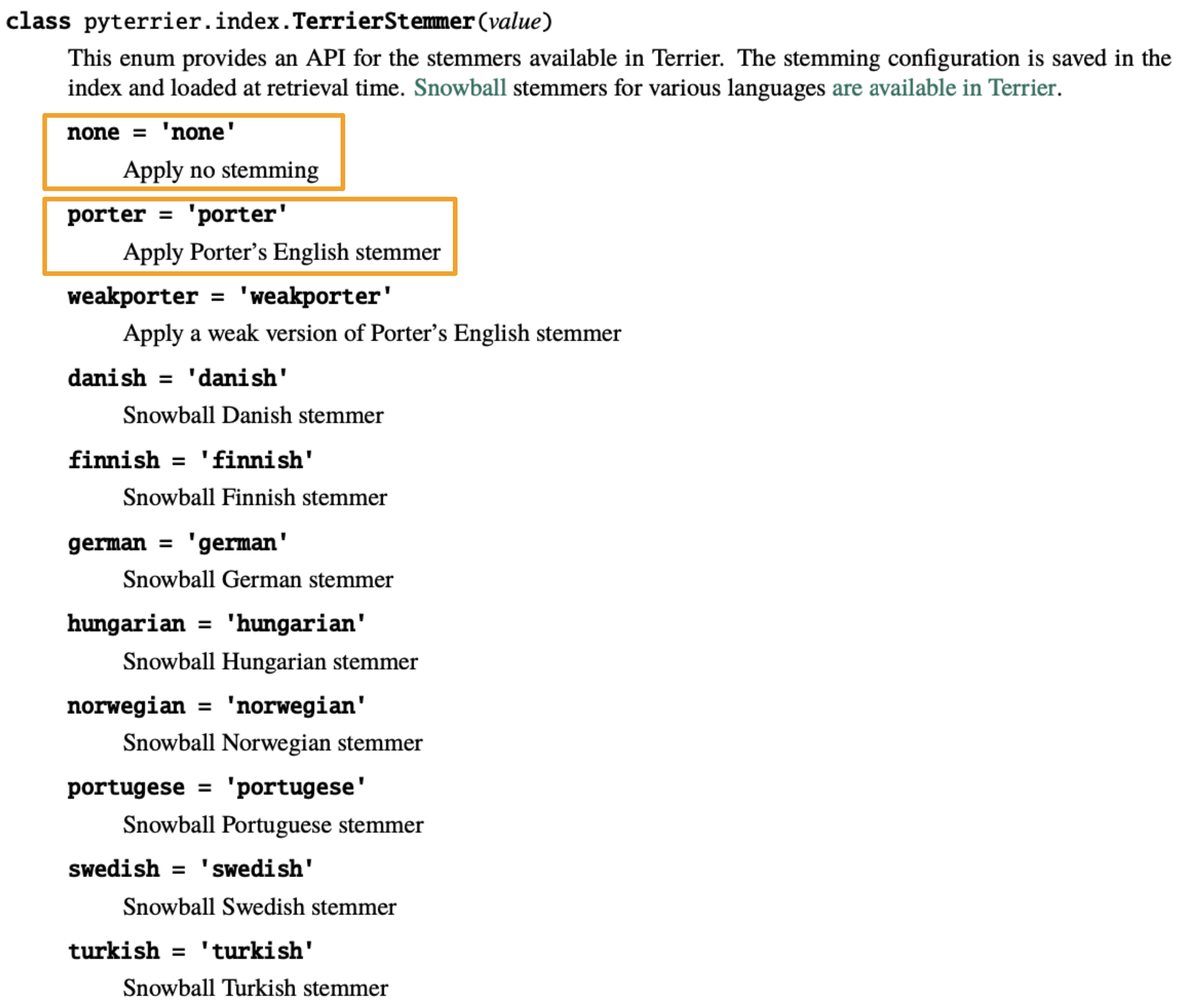
Two tokenization methods

Two tokenization methods



	recall_5	recall_10	P_5	P_10	ndcg
🟦 TF	0	0	0	0	4E-06
🟧 TF-IDF	0	0.000122	0	0.0005	1.2E-05
⬜ BM25	0	0	0	0	-7E-06

Stemming vs. no stemming



class pyterrier.index.**TerrierStemmer**(*value*)

This enum provides an API for the stemmers available in Terrier. The stemming configuration is saved in the index and loaded at retrieval time. Snowball stemmers for various languages are available in Terrier.

none = 'none'
Apply no stemming

porter = 'porter'
Apply Porter's English stemmer

weakporter = 'weakporter'
Apply a weak version of Porter's English stemmer

danish = 'danish'
Snowball Danish stemmer

finnish = 'finnish'
Snowball Finnish stemmer

german = 'german'
Snowball German stemmer

hungarian = 'hungarian'
Snowball Hungarian stemmer

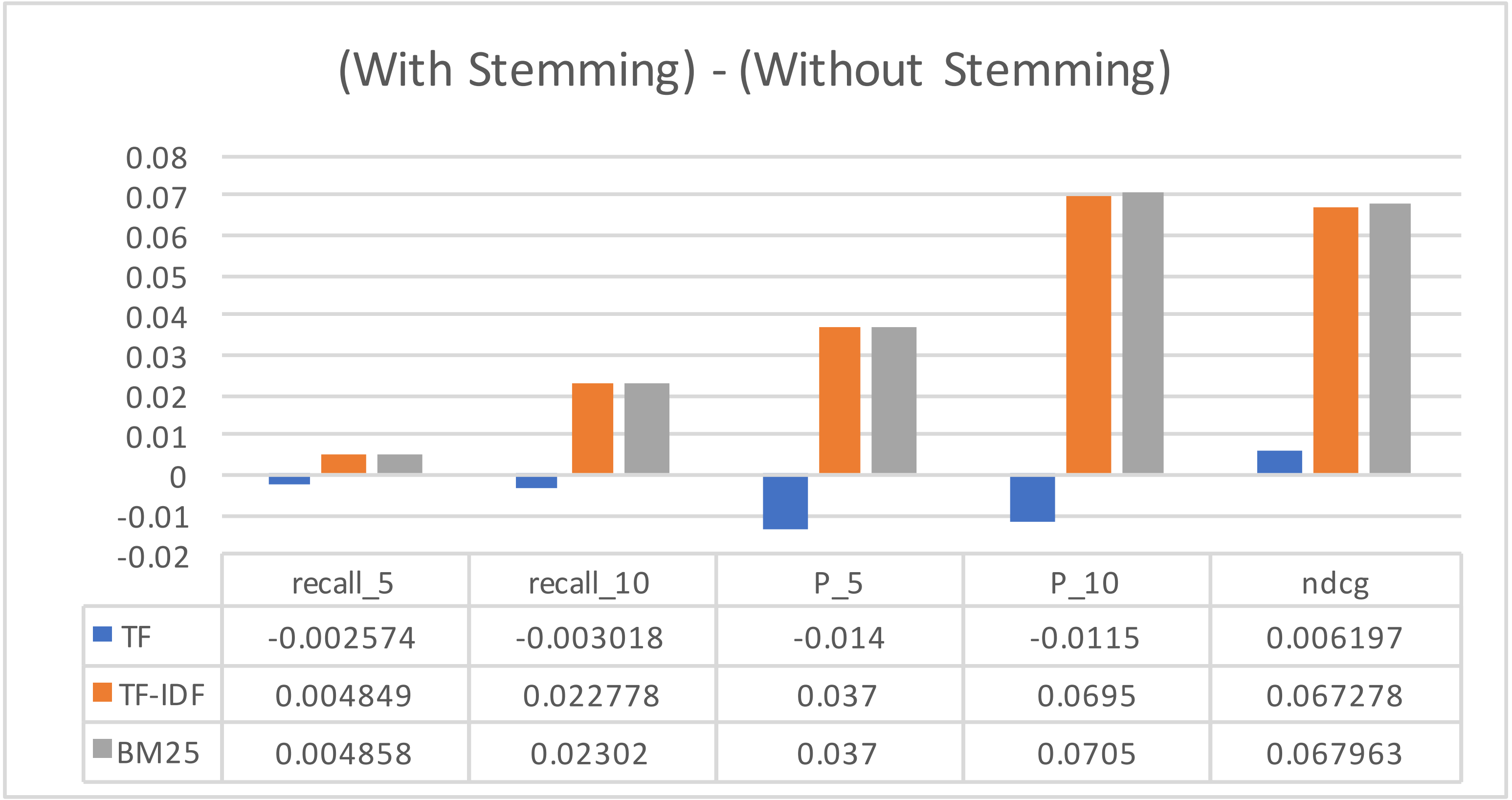
norwegian = 'norwegian'
Snowball Norwegian stemmer

portugese = 'portugese'
Snowball Portuguese stemmer

swedish = 'swedish'
Snowball Swedish stemmer

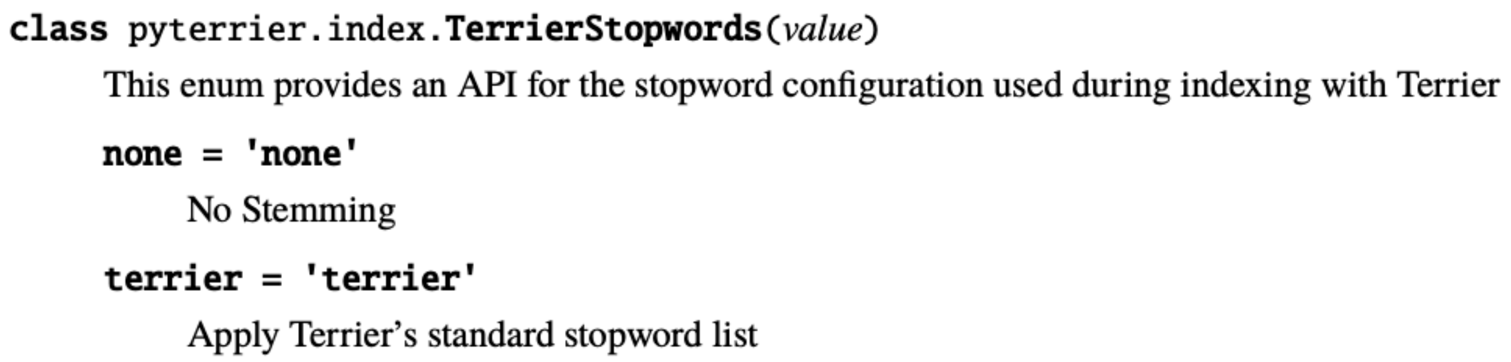
turkish = 'turkish'
Snowball Turkish stemmer

Stemming vs. no stemming



	recall_5	recall_10	P_5	P_10	ndcg
🟦 TF	-0.002574	-0.003018	-0.014	-0.0115	0.006197
🟧 TF-IDF	0.004849	0.022778	0.037	0.0695	0.067278
⬜ BM25	0.004858	0.02302	0.037	0.0705	0.067963

Stopword removal: yes vs. no



class pyterrier.index.**TerrierStopwords**(*value*)

This enum provides an API for the stopword configuration used during indexing with Terrier

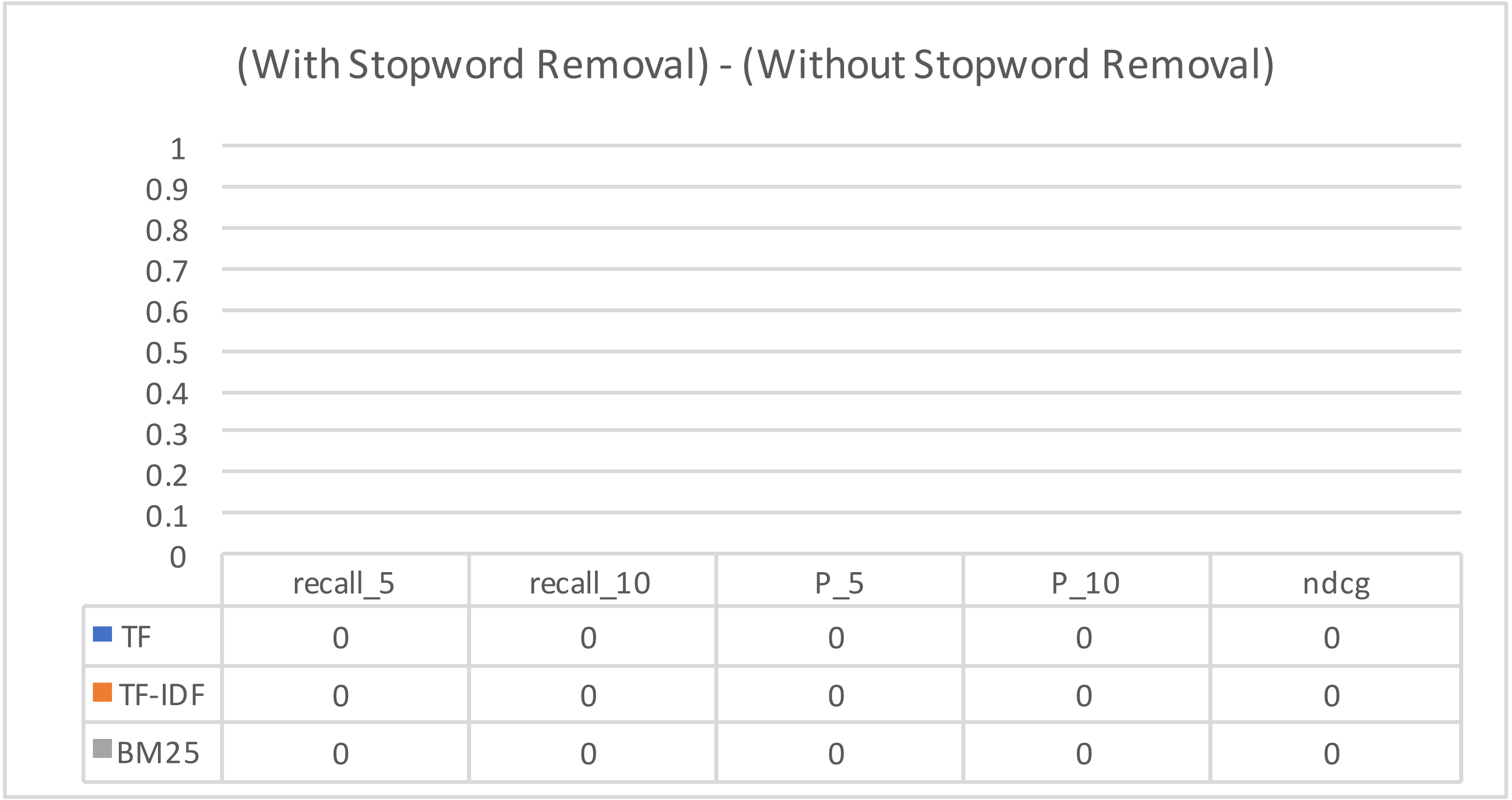
none = 'none'

No Stemming

terrier = 'terrier'

Apply Terrier’s standard stopword list

Stopword removal: yes vs. no



	recall_5	recall_10	P_5	P_10	ndcg
🟦 TF	0	0	0	0	0
🟧 TF-IDF	0	0	0	0	0
⬜ BM25	0	0	0	0	0

References

PyTerrier Documentation. (n.d.). Retrieved December 1, 2022, from https://pyterrier.readthedocs.io/_/downloads/en/latest/pdf/

Terrier-Org. (n.d.). *PyTerrier*. GitHub. Retrieved December 1, 2022, from https://github.com/terrier-org/pyterrier