Zhamilya Saparova

HW 4 - Neural Ranking

As a reference I used HW 2 Assignment. As in HW 2 Assignment, I created an index with the name “hello”. I wrote settings for the stemming with lowercase filter, “whitespace” tokenizer and snowball english analyzer. For the setting without stemming, I used only a lowercase filter and “whitespace” tokenizer.
Then, I recreated the index with proper settings. To index documents, I used parallel_bulk API. The results for setting with and without stemming are shown below:
With stemming:

```
CPU times: user 1min 19s, sys: 737 ms, total: 1min 20s
Wall time: 2min 54s
```

Without stemming:

```
CPU times: user 1min 18s, sys: 710 ms, total: 1min 19s
Wall time: 2min 4s
```

As a next step, I performed a query search. Firstly, I used the simplest query “match_all”, which takes nothing and returns all the documents. It took
With stemming:

```
CPU times: user 132 ms, sys: 7.32 ms, total: 139 ms
Wall time: 7.6 s
```

Without stemming:

```
CPU times: user 190 ms, sys: 9.1 ms, total: 199 ms
Wall time: 7.89 s
```

Then, I randomly took the phrase “private boats” and passed it as a query. It showed top 20 results for this query with scores. It took
With stemming:

```
CPU times: user 171 ms, sys: 6.51 ms, total: 178 ms
Wall time: 7.47 s
```

Without stemming:

```
CPU times: user 238 ms, sys: 8.86 ms, total: 247 ms
Wall time: 8.73 s
```

Further, I ran test queries and received top 20 results for each query and estimated query execution time. Below, query index, query execution time and top 20 results of 3 queries are shown (overall, there are 100 queries).

```
0     3.8543753623962402    dict_keys(['158491', '635537', '607552',
'1880296',  '1957435',  '360918',  '625257',  '742912',  '663828',
'1774491',  '2411344',  '1093529',  '685181',  '1956922',  '589549',
'2261272', '945068', '1158969', '1170039', '1180246'])
1     3.941791296005249      dict_keys(['5728',  '283099',  '98452',
'2149047',  '79940',  '1890681',  '592647',  '849904',  '501880',
'608495',  '645822',  '375146',  '1893219',  '970793',  '1750830',
'2184293', '911906', '255972', '1597007', '1404651'])
2     2.9725987911224365    dict_keys(['1354456', '13554', '506304',
'166298',  '1333624',  '2164978',  '1354099',  '2164385',  '2206256',
'1231923',  '1355603',  '1604564',  '107818',  '1343180',  '2108477',
'1355975', '1070', '1354754', '1212060', '1354324'])
```

Zhamilya Saparova

I used ir_measures to format qrels and runs. My runs:
With stemming + cosine similarity:
```
{AP: 0.00045862085630015234,
 P@10: 0.2419999999999999,
 P@20: 0.1640000000000001}
```
Without stemming + cosine similarity:
```
{AP: 0.00045862085630015234,
 P@10: 0.2419999999999999,
 P@20: 0.1640000000000001}
```

With stemming:
```
{P@10:   0.22599999999999987,   AP:   0.00040770588385999904,   P@20:
0.1640000000000001}
```
Without stemming:
```
{P@10:   0.22599999999999987,   AP:   0.0004083007253942143,   P@20:
0.16450000000000006}
```
Creator's runs:
```
{P@10:   0.3000000000000005,   AP:   0.010808749000016847,   P@20:
0.3000000000000005}
```

Below I am attaching link to my ipynb on github:
https://github.com/zhamilyaa/ir_assignment