# AIR Exercise 1 Report

Markus Gabriel 01326657
Johannes Vass 01327476

April 17, 2018

The source code of the project is publicly available on Github under `https://github.com/scriptator/advanced-information-retrieval-ss18`.

## 1 Execution

We implemented our search engine in python3 ($\geq 3.5$). The two main executable modules are of our prototype are `air18.index` and `air18.search`. Before being able to run the prototype the required python packages and other dependencies need to be installed with the following command (which uses pip) or directly via the file `requirements.txt`.

```
./install.sh
```

The indexing and search can then be run with

```
python3 -m air18.index
python3 -m air18.search
```

The required options are explained in the help output.

```
usage: python3 -m air18.index [-h] [--case-folding] [--stop-words]
                              [--stemming | --lemmatization] --indexing-method
                              {simple,spimi,map_reduce}
                              patterns [patterns ...]

positional arguments:
  patterns              One or several glob patterns matching directories
                        which contain original TREC or JSON files

optional arguments:
  -h, --help            show this help message and exit
  --case-folding        Apply case folding
  --stop-words          Remove stop words
  --stemming            Apply Porter Stemmer
  --lemmatization       Apply lemmatization from NLTK
  --indexing-method {simple,spimi,map_reduce}
                        Indexing method to use


usage: python3 -m air18.search [-h] [--topics-file TOPICS_FILE] [--show SHOW]
                               [--run-name RUN_NAME] [--topic TOPIC] [--debug]
                               {tf-idf,bm25,bm25va} ...

optional arguments:
  -h, --help            show this help message and exit
  --topics-file TOPICS_FILE, -t TOPICS_FILE
```

```
                        The topic file in TREC's format containing queries
  --show SHOW           Maximum number of documents to report per topic
  --run-name RUN_NAME   Arbitrary string which will be contained in the TREC
                        output as an identifier of the run
  --topic TOPIC         Query only for one given topic instead of all
  --debug, -d           Print debug output

similarity function:
  {tf-idf,bm25,bm25va}
    tf-idf              use TF-IDF
    bm25                use BM25
    bm25va              use BM25 Verboseness Fission Variant
```

# 2 Prototype description and Index Structure

Our index is saved as multiple files into the directory `~/.air18/index/`, which gets (re)created on every call to the indexer.
The settings file contains the command line parameters with which the index was created so that the search script knows with which settings the index was created and can therefore interpret it correctly.
The statistics file contains global statistical values required for the search. These are the number of documents, the average document length and the mean average term frequency.
The document statistics file contains the length and average term frequency of each document.
Then there are the actual index files containing the posting lists. For the simple variant the index file is only a dictionary of token values to the list of corresponding (docid, term frequency) tuples. For the map reduce variant there are multiple such index file each containing token values whose first characters are in the character range assigned to the file. The index_k.p file contains for example the posting lists of all tokens starting with characters between f and k. All the files mentioned up until now are in pickle format, allowing simple de-/serialization in python. The only files in a custom format are the intermediate and the final SPIMI indexes. These are text files and contain a line for each token posting. An example line is the following.

```
authorship:1230-1,2283-1,3787-1
```

Each value in front of the dash is a document id and each value after the dash the term frequency. The lines in the SPIMI files are ordered alphabetically, allowing the efficient merging of multiple intermediate SPIMI files into a final SPIMI file line by line. Since during search the whole SPIMI index would take very long to load (or would not event fit in memory at all), we also create a meta index over the final SPIMI index file. This index is a mapping of token values to the file position of their corresponding line in the index file. This meta index is again saved in pickle format.
Since the provided XML files were not well-formed, we created our own JSON versions of them which we used as input for the indexing.

For document processing we used simple strategies. Tokenization is done by

splitting on all non-alphanumeric characters with the exception of dots in words and the @ character to allow searching for emails and similar special strings. Case folding is performed by converting all words to lowercase. For stop word removal we relied on the stopword list of the python ntlk (natural language processing toolkit) library. We further used either the Porter Stemmer if stemming was chosen or the WordNetLemmatizer of the nltk library if lemmatization was chosen.

# 3 Score calculation

To show how the score is calculated for two documents adjacent in the ranked list, we used the query 404 "Ireland, peace talks" and queried using the BM25 scoring function. The first suggested document was "FBIS4-14340" and the second one "FBIS4-65646". The score is calculated as the sum of the following formula for each term.

```
B = (1 - b + b * dl / avgdl)
score = idf * tf * (k1 + 1) / (tf + k1 * B)
```

b, k and avgdl (average document length) are the same for all documents and terms. b and k had the default values of 0.25 and 1.5. The avgdl was 282.93275310414344 words. The idf (inverse document frequency) is different for each term but independent of the document. The idf of term "Ireland" is 4.365922300020734, of term "peace" 2.7571157861553934 and of term "talks" 2.0836810086844118. The dl (document length) of document "FBIS4-14340" was 281 and the dl of document "FBIS4-65646" was 2112. The tf (term frequency) of the terms in document "FBIS4-14340" is 12 for "Ireland", 8 for "peace" and 8 for "talks" and of the terms in document "FBIS4-65646" it is 46, 18 and 9. Inserting the values in the formula, we get the following results.

```
B1 = (1 - 0.25 + 0.25 * 281 / 282.93275310414344)

score1_1 = 4.365922300020734 * 12 * (1.5 + 1) / (12 + 1.5 * B1)
score1_2 = 2.7571157861553934 * 8 * (1.5 + 1) / (8 + 1.5 * B1)
score1_3 = 2.0836810086844118 * 8 * (1.5 + 1) / (8 + 1.5 * B1)
score1 = 19.897790841130274

B2 = (1 - 0.25 + 0.25 * 2112 / 282.93275310414344)
score2_1 = 4.365922300020734 * 46 * (1.5 + 1) / (46 + 1.5 * B2)
score2_2 = 2.7571157861553934 * 18 * (1.5 + 1) / (18 + 1.5 * B2)
score2_3 = 2.0836810086844118 * 9 * (1.5 + 1) / (9 + 1.5 * B2)
score2 = 19.34340445371477
```

# 4 Evaluation

The following table shows the MAP of the different similarity functions. The search was performed over an index created with case folding, stop words, and stemming activated. The results are as we expected. TF-IDF performs worst, then comes BM25, then the modified BM25 with Verboseness Fission. The modified BM25 automatically adjusts the k parameter based on the data so it

makes sense that it performs better. BM25 is more refined than TF-IDF and usually performs better, so it was no surprise that it also performed better in our case.

The significance test showed that for threshold 0.05 the null hypothesis for TF-IDF and BM25 (the distributions are the same) can be rejected with a p value of 1.0556001252449993e-08. The null hypothesis for TF-IDF and BM25VA can also be rejected with a p value of 1.6844221256191453e-07. The null hypothesis can, however, not be rejected for BM25 and BM25VA with a p value of 0.3993278520824677. That is, given the chosen parameters and our implementation, there is about 40% chance that the slightly better performance of BM25VA which we measured is only an effect of chance. As this is way beyond the threshold of 5%, we need to reject the hypothesis.

| | **TF-IDF** | **BM25** | **BM25VA** | | **TF-IDF** | **BM25** | **BM25VA** |
|---|---|---|---|---|---|---|---|
| 401 | 0.0174 | 0.0385 | 0.0488 | 427 | 0.1016 | 0.1293 | 0.1400 |
| 402 | 0.1004 | 0.1599 | 0.1435 | 428 | 0.1234 | 0.2129 | 0.2142 |
| 403 | 0.7474 | 0.7506 | 0.7247 | 429 | 0.1832 | 0.3370 | 0.3995 |
| 404 | 0.1136 | 0.1730 | 0.1707 | 430 | 0.5258 | 0.5610 | 0.6272 |
| 405 | 0.0306 | 0.0512 | 0.0494 | 431 | 0.3278 | 0.5064 | 0.5155 |
| 406 | 0.2711 | 0.4213 | 0.3840 | 432 | 0.0051 | 0.0014 | 0.0012 |
| 407 | 0.1429 | 0.2531 | 0.2464 | 433 | 0.0022 | 0.0238 | 0.0248 |
| 408 | 0.0653 | 0.1239 | 0.1377 | 434 | 0.2855 | 0.3924 | 0.3908 |
| 409 | 0.0079 | 0.0922 | 0.1427 | 435 | 0.1154 | 0.0675 | 0.0680 |
| 410 | 0.7142 | 0.8612 | 0.8613 | 436 | 0.0060 | 0.0455 | 0.0591 |
| 411 | 0.0798 | 0.1602 | 0.1755 | 437 | 0.0396 | 0.0271 | 0.0257 |
| 412 | 0.0229 | 0.1731 | 0.2608 | 438 | 0.1796 | 0.2358 | 0.2335 |
| 413 | 0.1279 | 0.0919 | 0.0786 | 439 | 0.0139 | 0.0295 | 0.0308 |
| 414 | 0.0878 | 0.2304 | 0.2404 | 440 | 0.0134 | 0.0550 | 0.0739 |
| 415 | 0.1256 | 0.2201 | 0.2280 | 441 | 0.4261 | 0.6355 | 0.5902 |
| 416 | 0.2461 | 0.3478 | 0.3371 | 442 | 0.0049 | 0.0116 | 0.0094 |
| 417 | 0.2375 | 0.3091 | 0.2995 | 443 | 0.0474 | 0.1524 | 0.1585 |
| 418 | 0.0635 | 0.3738 | 0.3565 | 444 | 0.4000 | 0.5560 | 0.5816 |
| 419 | 0.0973 | 0.0571 | 0.0507 | 445 | 0.0819 | 0.1569 | 0.1759 |
| 420 | 0.2230 | 0.3281 | 0.3890 | 446 | 0.0967 | 0.1672 | 0.1837 |
| 421 | 0.0369 | 0.0412 | 0.0422 | 447 | 0.2409 | 0.2634 | 0.0974 |
| 422 | 0.0617 | 0.2251 | 0.2948 | 448 | 0.0048 | 0.0183 | 0.0227 |
| 423 | 0.6270 | 0.6159 | 0.5949 | 449 | 0.1111 | 0.0914 | 0.0850 |
| 424 | 0.1129 | 0.1384 | 0.1349 | 450 | 0.1795 | 0.2833 | 0.2994 |
| 425 | 0.2904 | 0.4260 | 0.4360 | **avg** | **0.1636** | **0.2331** | **0.2374** |
| 426 | 0.0109 | 0.0335 | 0.0359 | | | | |