Assignment-1

Data Preprocessing:

Data Extraction:

We have done the relevant text preprocessing by extracting the title and text, concatenating both fields. The data from tags is extracted using the BeautifulSoup library. We have performed the same for all the 1400 files and shown the text before and after the preprocessing for 5 such samples in our code file. Here, we are showing the output for one such file.

Before extraction:

```
Before Extraction :  <doc>
<docno>
1011
</docno>
<title>
free-flight measurements of the static and dynamic
</title>
<author>
</author>
<biblio>
</biblio>
<text>
  charts have been prepared relating the thermodynamic properties of
air in chemical equilibrium for temperatures to 15,000 k and for pressures
from 10 to 10 atmospheres .  also included are charts showing
the composition of air, the isentropic exponent, and the speed of
sound .  these charts are based on thermodynamic data calculated by the
national bureau of standards .
</text>
</doc>
```

After extraction:

```
After Extraction :
free-flight measurements of the static and dynamic

  charts have been prepared relating the thermodynamic properties of
air in chemical equilibrium for temperatures to 15,000 k and for pressures
from 10 to 10 atmospheres .  also included are charts showing
the composition of air, the isentropic exponent, and the speed of
sound .  these charts are based on thermodynamic data calculated by the
national bureau of standards .
```

Preprocessing:

We have done the preprocessing specified in the assignment in the mentioned order:-

1. Convert the text to lowercase.
2. Perform tokenization on lowercase text input using nltk library.
3. Removed stopwords from tokenized dataset using nltk library.
4. Removed punctuation using string library.
5. Removed leftover blank space tokens.

We have performed the above steps on the entire dataset, and below screenshot shows the text files before and after preprocessing.

```
Before lower case text :
the operation of the npl 18in x 14in. wind tunnel in the transonic speed
 range .

a brief description of the slotted liners used is given together with
the power requirements and some flow surveys .
some observations are made on wall interference on a half-model of a
swept wing tested in the wind tunnel .

After lower case text and before tokenization :
the operation of the npl 18in x 14in. wind tunnel in the transonic speed
 range .

a brief description of the slotted liners used is given together with
the power requirements and some flow surveys .
some observations are made on wall interference on a half-model of a
swept wing tested in the wind tunnel .

After tokenization and before stopwords removal :  ['the', 'operation', 'of', 'the', 'npl', '18in', 'x', '14in', '.', 'wind', 'tunnel', 'in', 'the', 't
ransonic', 'speed', 'range', '.', 'a', 'brief', 'description', 'of', 'the', 'slotted', 'liners', 'used', 'is', 'given', 'together', 'with', 'the', 'pow
er', 'requirements', 'and', 'some', 'flow', 'surveys', '.', 'some', 'observations', 'are', 'made', 'on', 'wall', 'interference', 'on', 'a', 'half-mode
l', 'of', 'a', 'swept', 'wing', 'tested', 'in', 'the', 'wind', 'tunnel', '.']
After stopwords removal and before punctuations removal :  ['operation', 'npl', '18in', 'x', '14in', '.', 'wind', 'tunnel', 'transonic', 'speed', 'rang
e', '.', 'brief', 'description', 'slotted', 'liners', 'used', 'given', 'together', 'power', 'requirements', 'flow', 'surveys', '.', 'observations', 'ma
de', 'wall', 'interference', 'half-model', 'swept', 'wing', 'tested', 'wind', 'tunnel', '.']
After punctuations and before blank space token removal :  ['operation', 'npl', '18in', 'x', '14in', 'wind', 'tunnel', 'transonic', 'speed', 'range',
'brief', 'description', 'slotted', 'liners', 'used', 'given', 'together', 'power', 'requirements', 'flow', 'surveys', 'observations', 'made', 'wall',
'interference', 'half-model', 'swept', 'wing', 'tested', 'wind', 'tunnel']
After blank space token removal :  ['operation', 'npl', '18in', 'x', '14in', 'wind', 'tunnel', 'transonic', 'speed', 'range', 'brief', 'description',
'slotted', 'liners', 'used', 'given', 'together', 'power', 'requirements', 'flow', 'surveys', 'observations', 'made', 'wall', 'interference', 'half-mod
el', 'swept', 'wing', 'tested', 'wind', 'tunnel']
```

Boolean Query and Unigram Inverted Index:

We have created the unigram inverted index from scratch on our dataset after performing all the preprocessing steps. We have used the pickle module to save the unigram inverted index.

Our boolean query on the dataset supports all the mentioned operations as well as the generalized query and supports the calculation of the minimum number of comparisons for each of the operations.

Methodology:

- Initialized empty unigram dictionary.
- Using the tokenized list of all the docs, we iterated through each doc.
- For each document and each token check if token exists in the document or not.
- If it exists, append the doc index to the list corresponding to token.
- During testing, preprocess the input in same way as done during dictionary creation.
- Processed input is passed through the boolean queries using OR, AND, OR NOT, AND NOT.
- The query processing takes couple of unigram tokens and a boolean operator.
- The list of documents for the query is displayed.

Results:

```
Enter the number of queries.3
Query : shock wave from the surface
Operand : AND,AND
Query : very high speed flight
Operand : AND,OR
Query : velocity and temperature profiles
Operand : AND,AND NOT
Query 1:  shock AND wave AND surface
Number of documents retrieved for query 1:  27
Names of the documents retrieved for query 1:  ['cranfield0334', 'cranfield1248', 'cranfield1107', 'cranfield0612',
'cranfield1198', 'cranfield0072', 'cranfield1367', 'cranfield1356', 'cranfield0665', 'cranfield0800', 'cranfield032
9', 'cranfield0504', 'cranfield0798', 'cranfield0192', 'cranfield0569', 'cranfield0415', 'cranfield1391', 'cranfield0
025', 'cranfield1307', 'cranfield1300', 'cranfield1151', 'cranfield1364', 'cranfield1313', 'cranfield0439', 'cranfiel
d0110', 'cranfield0976', 'cranfield1267']
Number of comparisons required for query 1:  624
```

```
Query 2:  high AND speed OR flight
Number of documents retrieved for query 2:  153
Names of the documents retrieved for query 2:  ['cranfield0553', 'cranfield0302', 'cranfield0792', 'cranfield0759',
'cranfield0933', 'cranfield0101', 'cranfield0969', 'cranfield1279', 'cranfield1089', 'cranfield0369', 'cranfield016
4', 'cranfield1270', 'cranfield0163', 'cranfield1213', 'cranfield0758', 'cranfield0162', 'cranfield1043', 'cranfield1
271', 'cranfield0198', 'cranfield0395', 'cranfield1247', 'cranfield0968', 'cranfield0214', 'cranfield1303', 'cranfiel
d0813', 'cranfield0878', 'cranfield0876', 'cranfield0429', 'cranfield1154', 'cranfield0624', 'cranfield0416', 'cranfi
eld0812', 'cranfield1101', 'cranfield0870', 'cranfield0625', 'cranfield1356', 'cranfield1163', 'cranfield0067', 'cran
field0058', 'cranfield0252', 'cranfield1380', 'cranfield0290', 'cranfield1170', 'cranfield0069', 'cranfield1183', 'cr
anfield0051', 'cranfield0263', 'cranfield0435', 'cranfield0899', 'cranfield0806', 'cranfield0230', 'cranfield0033',
'cranfield0606', 'cranfield1343', 'cranfield0253', 'cranfield1381', 'cranfield1147', 'cranfield0456', 'cranfield080
9', 'cranfield1321', 'cranfield0209', 'cranfield1310', 'cranfield0981', 'cranfield0185', 'cranfield0986', 'cranfield1
095', 'cranfield0729', 'cranfield0917', 'cranfield0319', 'cranfield0141', 'cranfield0374', 'cranfield1299', 'cranfiel
d0945', 'cranfield1263', 'cranfield0311', 'cranfield0329', 'cranfield1002', 'cranfield1077', 'cranfield0169', 'cranfi
eld0700', 'cranfield0352', 'cranfield0798', 'cranfield1040', 'cranfield0962', 'cranfield0505', 'cranfield1243', 'cran
field0309', 'cranfield0593', 'cranfield1217', 'cranfield0556', 'cranfield0629', 'cranfield1161', 'cranfield0275', 'cr
anfield1157', 'cranfield0085', 'cranfield0049', 'cranfield0076', 'cranfield0082', 'cranfield1104', 'cranfield0441',
'cranfield0025', 'cranfield0810', 'cranfield0014', 'cranfield1300', 'cranfield0077', 'cranfield1169', 'cranfield028
0', 'cranfield1158', 'cranfield0012', 'cranfield0476', 'cranfield0216', 'cranfield0024', 'cranfield1102', 'cranfield0
681', 'cranfield0211', 'cranfield0657', 'cranfield0805', 'cranfield1324', 'cranfield1111', 'cranfield0037', 'cranfiel
d0453', 'cranfield1377', 'cranfield0860', 'cranfield0052', 'cranfield0436', 'cranfield1379', 'cranfield0293', 'cranfi
eld0036', 'cranfield0204', 'cranfield1110', 'cranfield0804', 'cranfield0803', 'cranfield0430', 'cranfield0292', 'cran
field0401', 'cranfield1188', 'cranfield0574', 'cranfield0746', 'cranfield0545', 'cranfield1000', 'cranfield0172', 'cr
anfield0516', 'cranfield1295', 'cranfield0976', 'cranfield0378', 'cranfield0982', 'cranfield0143', 'cranfield0715',
'cranfield1065', 'cranfield0581', 'cranfield1063', 'cranfield0948', 'cranfield1294']
Number of comparisons required for query 2:  405
```

```
Query 3:  velocity AND temperature AND NOT profiles
Number of documents retrieved for query 3:  55
Names of the documents retrieved for query 3:  ['cranfield0562', 'cranfield1215', 'cranfield0131', 'cranfield0304',
'cranfield1010', 'cranfield1222', 'cranfield0966', 'cranfield0959', 'cranfield0154', 'cranfield1240', 'cranfield107
2', 'cranfield1335', 'cranfield0646', 'cranfield0043', 'cranfield0081', 'cranfield1366', 'cranfield0072', 'cranfield0
481', 'cranfield1393', 'cranfield0073', 'cranfield1199', 'cranfield0051', 'cranfield0050', 'cranfield1149', 'cranfiel
d0059', 'cranfield0236', 'cranfield0460', 'cranfield0944', 'cranfield0773', 'cranfield0549', 'cranfield0169', 'cranfi
eld0352', 'cranfield0550', 'cranfield0962', 'cranfield0611', 'cranfield0872', 'cranfield1157', 'cranfield0414', 'cran
field1301', 'cranfield0695', 'cranfield0097', 'cranfield0407', 'cranfield0269', 'cranfield0260', 'cranfield0267', 'cr
anfield0268', 'cranfield0062', 'cranfield0406', 'cranfield1204', 'cranfield0325', 'cranfield0126', 'cranfield0949',
'cranfield0378', 'cranfield1268', 'cranfield0383']
Number of comparisons required for query 3:  1753
```

Bigram Inverted Index:

Methodology:

- Initialized empty bigram dictionary.
- Using the tokenized list of all the docs, we iterated through each doc.
- For each doc picked consecutive tokens and create a bigram phrase : “token1 token2”.
- Checked if the phrase “token1 token2” exists in the dictionary or not.
- If it does not, created the key-value pair otherwise appended the doc index to the list corresponding to phrase key.
- During testing, processed the input in same way as done during dictionary creation.
- Created bigram phrases by selecting consecutive tokens. Created a list of operand consisting of AND operator of length one less than that of length of bigram phrases.

Positional Index:

We have created the positional indexing from scratch and used the pickle module to save the index.

Methodology:

We have created a dictionary of dictionary to store the positional index of each of the tokens that occur across the dataset. For each of the tokens, we store the documents and the token's position within the document using a dictionary structure. To query a phase, we will first preprocess the input as we have preprocessed our dataset to tokenize the query. Then fixing the first token of the query, we will extract the document number and position of the token. Then we will check if all the subsequent token occurs at a consecutive position within the same document to fetch all the relevant documents.

Assumptions: We assume that the phrasal query will contain at least two words.

Results of phrase queries:

```
Enter the number of queries.3
Query : shock wave from the surface
Query : very high speed flight
Query : velocity and temperature profiles
['shock wave', 'wave surface']
['high speed', 'speed flight']
['velocity temperature', 'temperature profiles']
Number of documents retrieved for query 1 using bigram inverted index:  2
Names of documents retrieved for query  1 using bigram inverted index:  ['cranfield1151', 'cranfield0110']
Number of documents retrieved for query 1 using positional index:  2
Names of documents retrieved for query  1 using positional index:  ['cranfield1151', 'cranfield0110']
Number of documents retrieved for query 2 using bigram inverted index:  5
Names of documents retrieved for query  2 using bigram inverted index:  ['cranfield0606', 'cranfield0700', 'cranfield
0012', 'cranfield0024', 'cranfield0746']
Number of documents retrieved for query 2 using positional index:  5
Names of documents retrieved for query  2 using positional index:  ['cranfield0606', 'cranfield0700', 'cranfield001
2', 'cranfield0024', 'cranfield0746']
Number of documents retrieved for query 3 using bigram inverted index:  6
Names of documents retrieved for query  3 using bigram inverted index:  ['cranfield0967', 'cranfield1109', 'cranfield
0088', 'cranfield0061', 'cranfield0328', 'cranfield0071']
Number of documents retrieved for query 3 using positional index:  6
Names of documents retrieved for query  3 using positional index:  ['cranfield0967', 'cranfield1109', 'cranfield008
8', 'cranfield0061', 'cranfield0328', 'cranfield0071']
```

The phrasal query generates the same output using the bigram inverted indexing and positional indexing for the above 4 queries.