# MSc Data Mining and Machine Learning (2019)

# Lab 1 – Text Retrieval

|  |  |  |
| --- | --- | --- |
| Instructor: |  | Dr Peter Jancovic |
| Written by: |  | James J Nkhata |
|  |  |  |
| Experiment Date: |  | 04/02/2019 |
| Report Delivery Date: |  | 18/02/2019 |

## PART 1: TF-IDF BASED TEXT RETRIEVAL

### Purpose

The aim of the lab was to apply a list of text-based Information Retrieval (IR) techniques to a corpus of 112 documents (contained within a given folder “beng”) and give an analysis of the results obtained. The techniques applied were: Stop-word removal; Stemming; Construction of the index and Retrieval.

### Procedure

* + 1. Compiled the provided C files (.c) to executable files (.exe) using Microsoft Visual Studio Command prompt. The files provided for the Lab were stop.c, porter-stemmer.c, index.c and retrieve.c (C files):
       1. At the Microsoft Visual Studio command prompt typing ‘cl stop.c <ENTER>’ generated stop.exe
       2. ‘cl porter-stemmer.c <ENTER>’ generated porter-stemmer.exe
       3. ‘cl index.c <ENTER>’ generated index.exe
       4. ‘cl retrieve.c <ENTER>’ generated retrieve.exe
    2. Applied Stop-word removal to the 112 documents in the corpus:
       1. Created a folder “stop” within the same directory holding the corpus of documents “beng”
       2. Ran the provided stopScript.bat (script file). The script used the given stoplist50 to run the command that was equivalent to ‘stop stoplist50 beng\Abassim.txt > stop\AbassiM.stp’ for every document within the “beng” folder and saved its corresponding stop (.stp) output file into the “stop” folder
       3. Copied and pasted the contents of AgricoleW.txt file into Microsoft Word to get a read of the word count it contained. Repeated the process for AgricoleW.stp file.
    3. Applied Porter stemmer to the files contained in the “stop” folder:
       1. Created a folder “stem” within the same directory with “beng” and “stop” folders
       2. Ran the stemScript.bat that was also provided. Similar to the stopScript.bat, the stemScript ran the equivalence to the command ‘porter-stemmer stop \AbassiM.stp > stem\AbassiM.stm’ for every document within the “stop” folder and saved its corresponding stem (.stm) output file into the “stm” folder
       3. Opened the contents of AgricoleW.stm, observed the effects of the stemmer on the words “communications, sophisticated and transmissions” compared to the results to the file AgricoleW.stp
    4. Created document index files from the “beng”, “stop” and “stem” folders:
       1. Using the index.exe and a given file textFileList (which contained a list of all the text files (.txt) found in “beng” folder), ran the command ‘index textFileList > textIndex’ to produce the ‘textIndex’ file
       2. Used the given file stopFileList (which contained a list of all the .stp files found in the “stop” folder), ran the command ‘index stopFileList > stopIndex’ to produce the ‘stopIndex’ file
       3. Used the given file stemFileList (which contained a list of all the .stm files found in the “stop” folder), ran the command ‘index stemFileList > stemIndex’ to produce the ‘stemIndex’ file
       4. Analysed the produced index files in respect to DongP.txt, DongP.stp, DongP.stm, as well as the terms “adjacent” and “algorithm”
    5. Retrieval:
       1. Created a ‘query’ file with the terms “communication and networks” using a text editor
       2. Using retrieve.exe, ran the command ‘retrieve textIndex query > textIndexQueryResult’ to save the contents of the retrieve process into a file called textIndexQueryResult
       3. Applied the stop-word removal to the ‘query’ file using the command ‘stop stopList50 query > query.stp’. Ran the command ‘retrieve stopIndex query.stp > stopIndexQueryResult’ to save the contents of the retrieve process into a file called StopIndexQueryResult
       4. Applied the porter-stemmer to the ‘query.stp’ file using the command ‘porter-stemmer query.stp > query.stm’. Ran the command ‘retrieve query.stm > stemIndexQueryResult’ to save the contents of the retrieve process into a file called StemIndexQueryResult
       5. Repeated step 1.2.5. for query” algorithm and adjacent”.

### Experimental data

* + 1. Task 1 & 2: Compiling C files to EXE

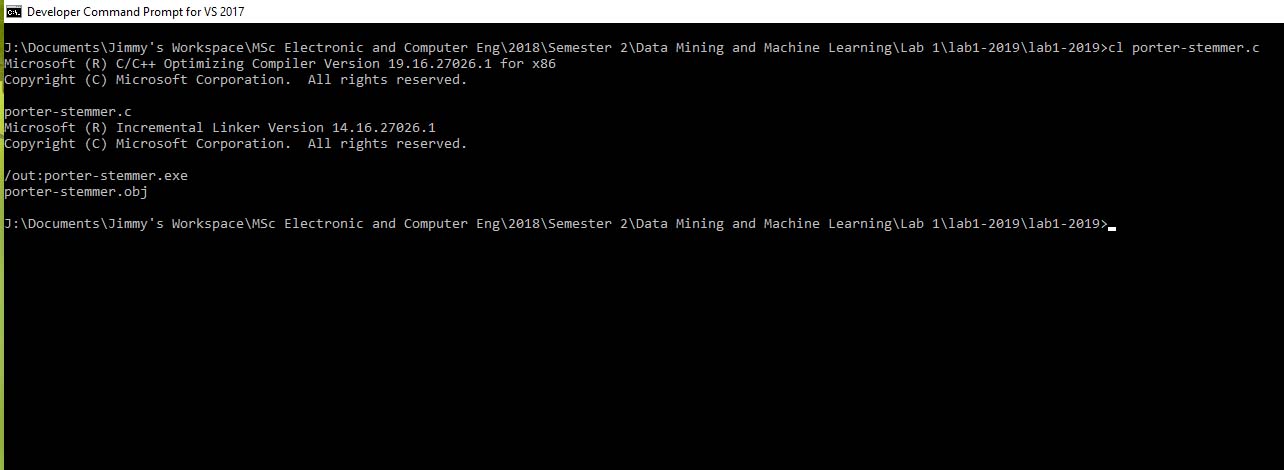


Figure 1: Command Prompt Compilation of porter-stemmer.c (Microsoft Visual Studio 2017 )

* + 1. Task 3: Stop-word Removal

AgricoleW.txt word count in Microsoft Word = 374 words

AgricoleW.stp word count in Microsoft Word = 242 words

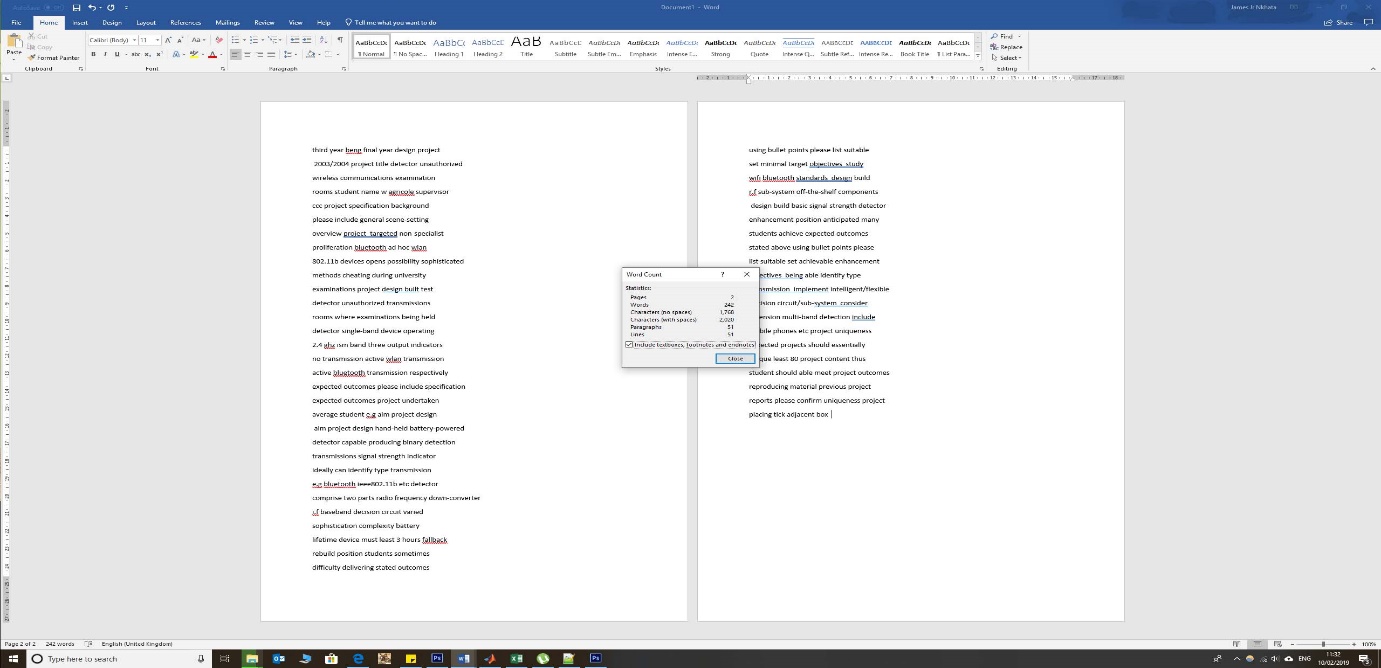


Figure 2: Word count of AgricoleW.stp (topped AgricoleW.txt)

* + 1. Task 4: Stemming

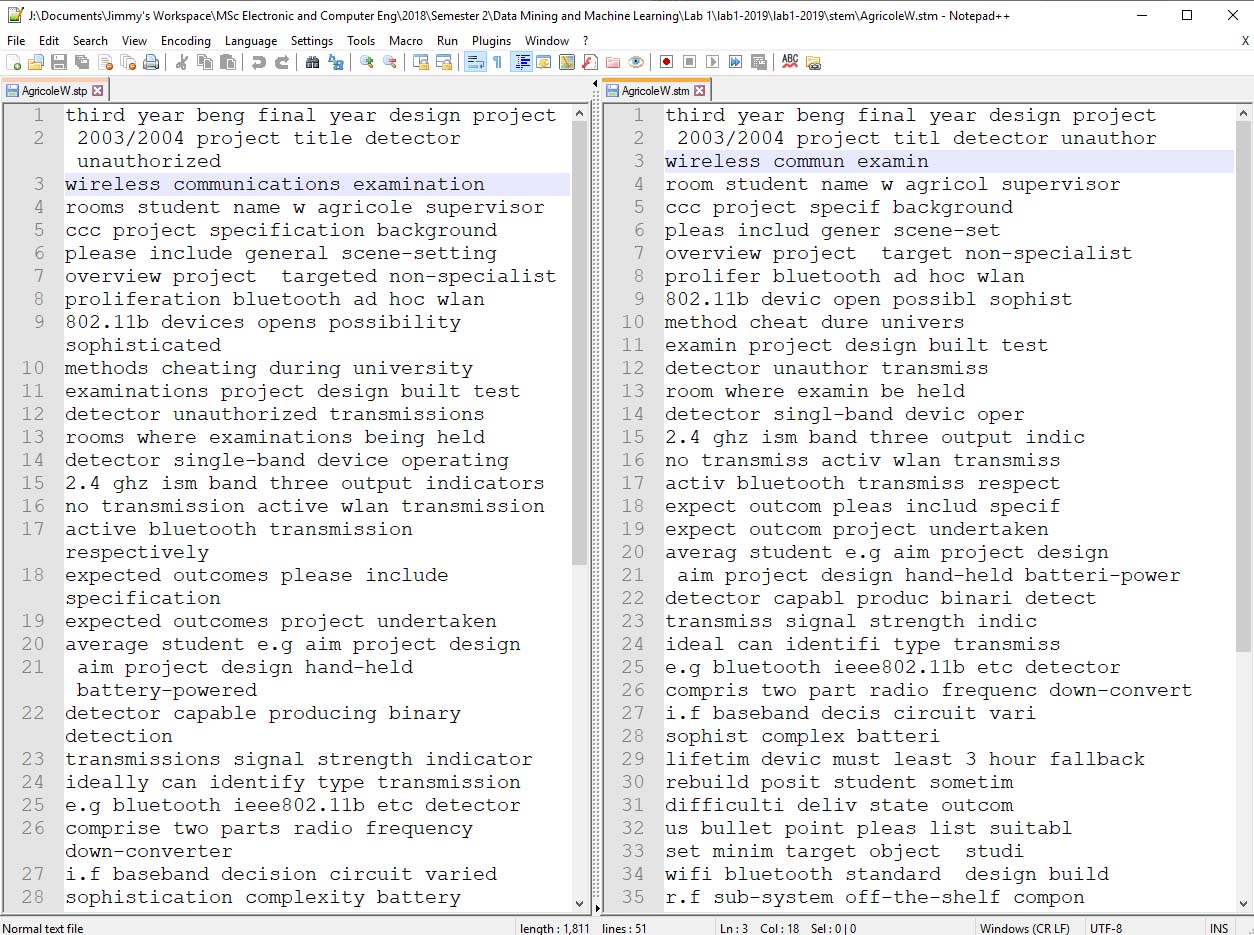


Figure 3: Side-by-side comparison of AgricoleW.stp and AgricoleW.stm

* + 1. Task 5: Creating Document Index Files

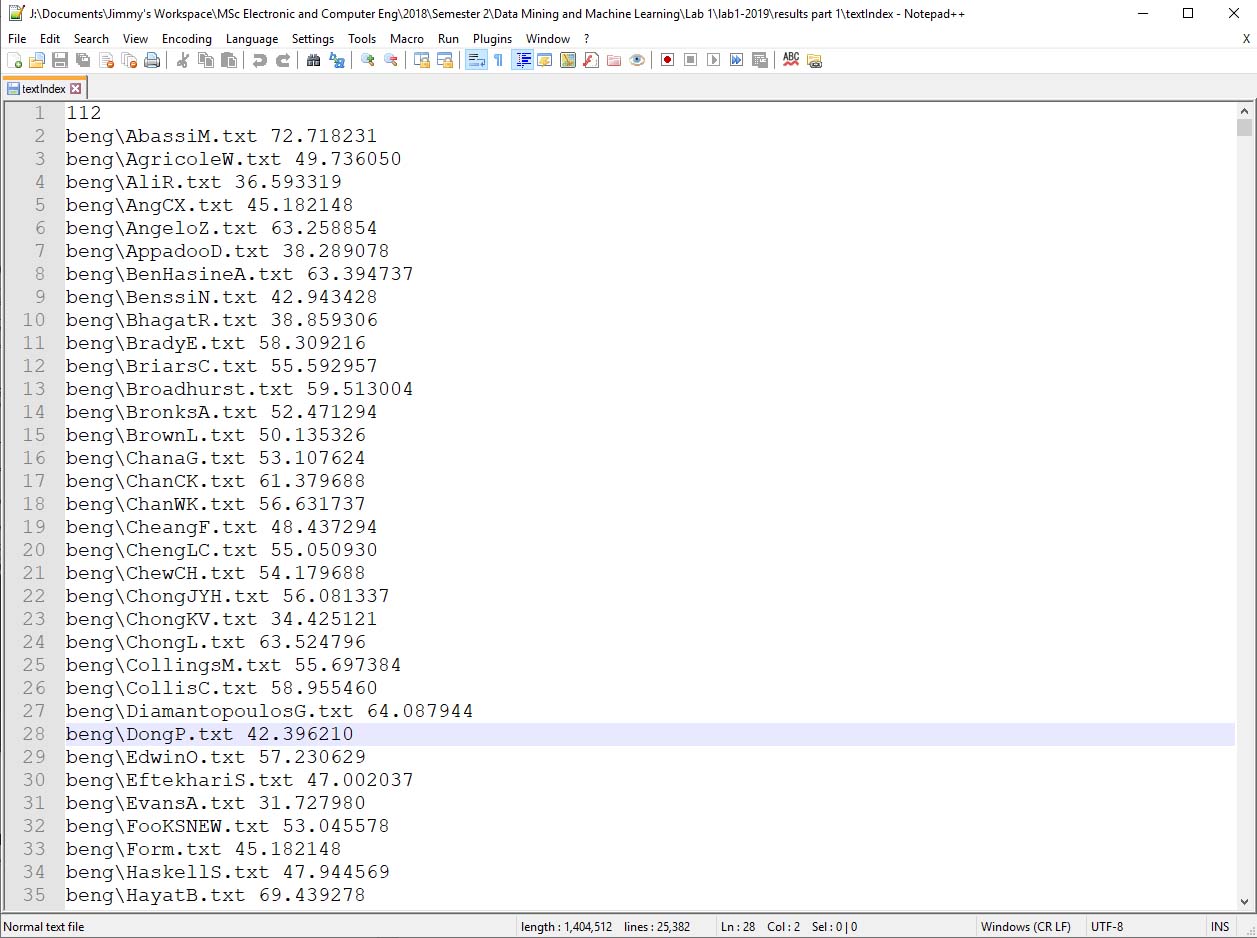


Figure 4: textIndex file

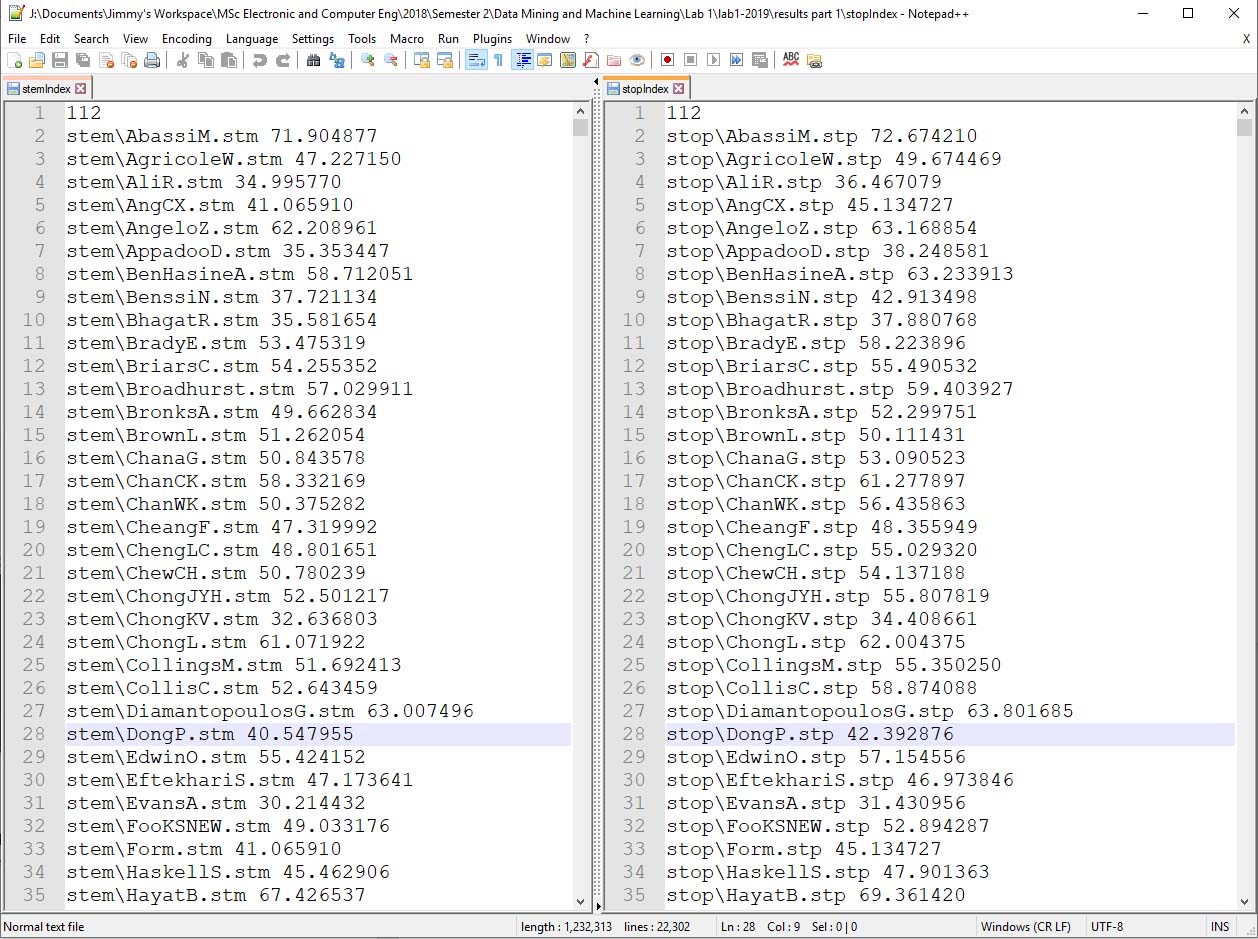


Figure 5: Side-by-side stopIndex and stemIndex files

* + 1. Task 6 & 7: Retrieval

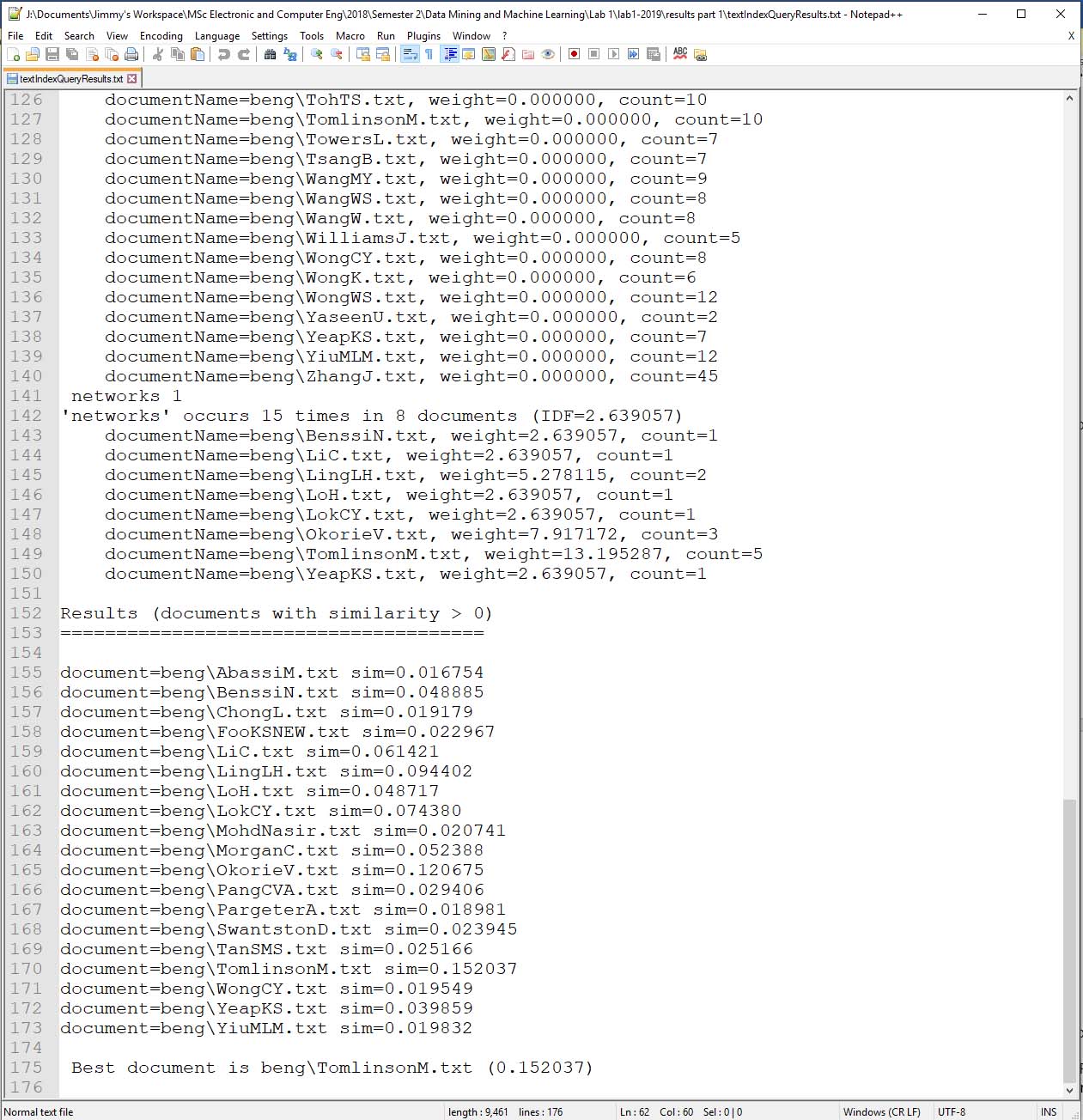


Figure 6: Retrieve on textIndex file using query (communication and networks)

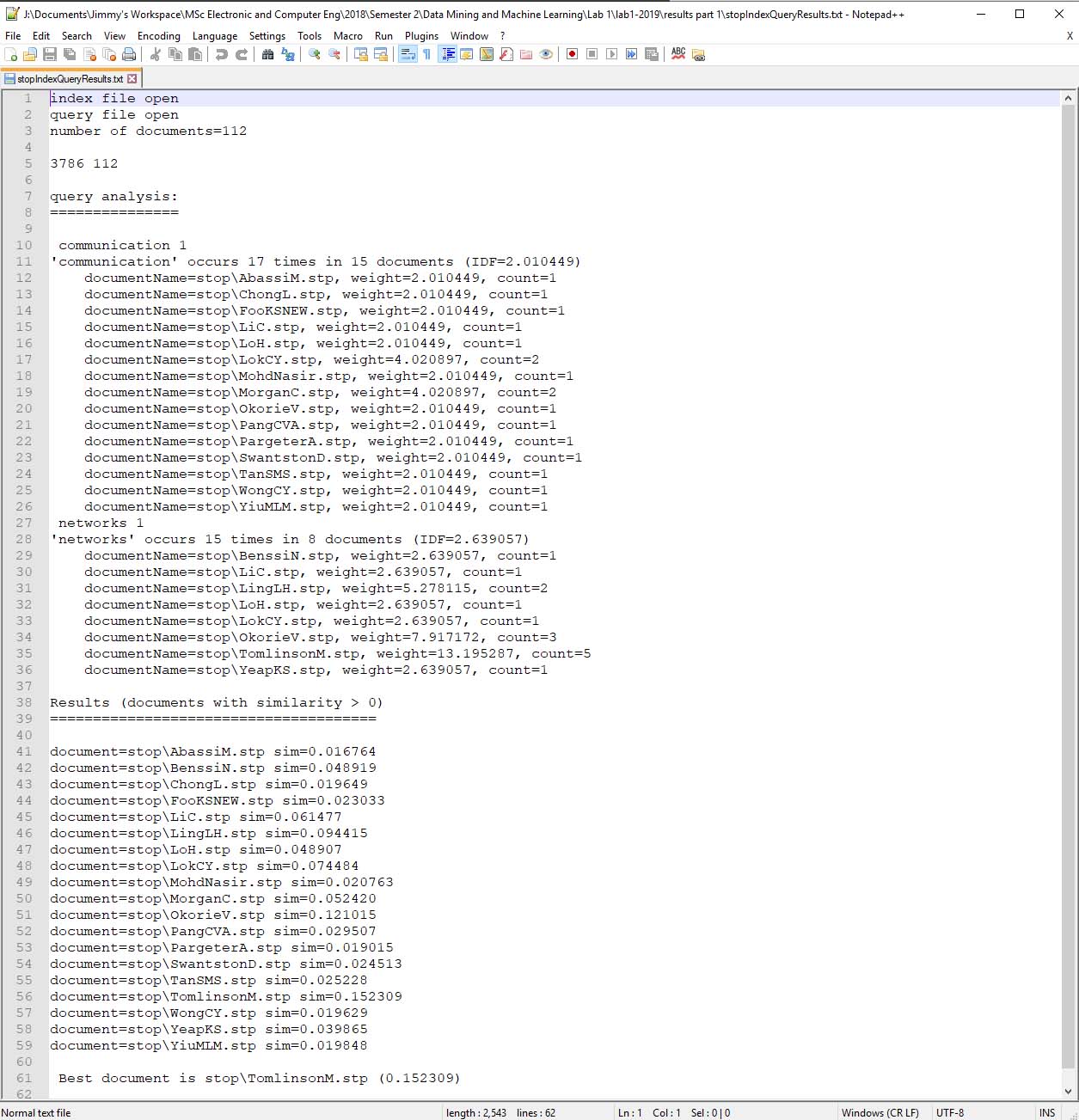


Figure 7: Retrieve on stopIndex file using query.stp (query with stopping applied)

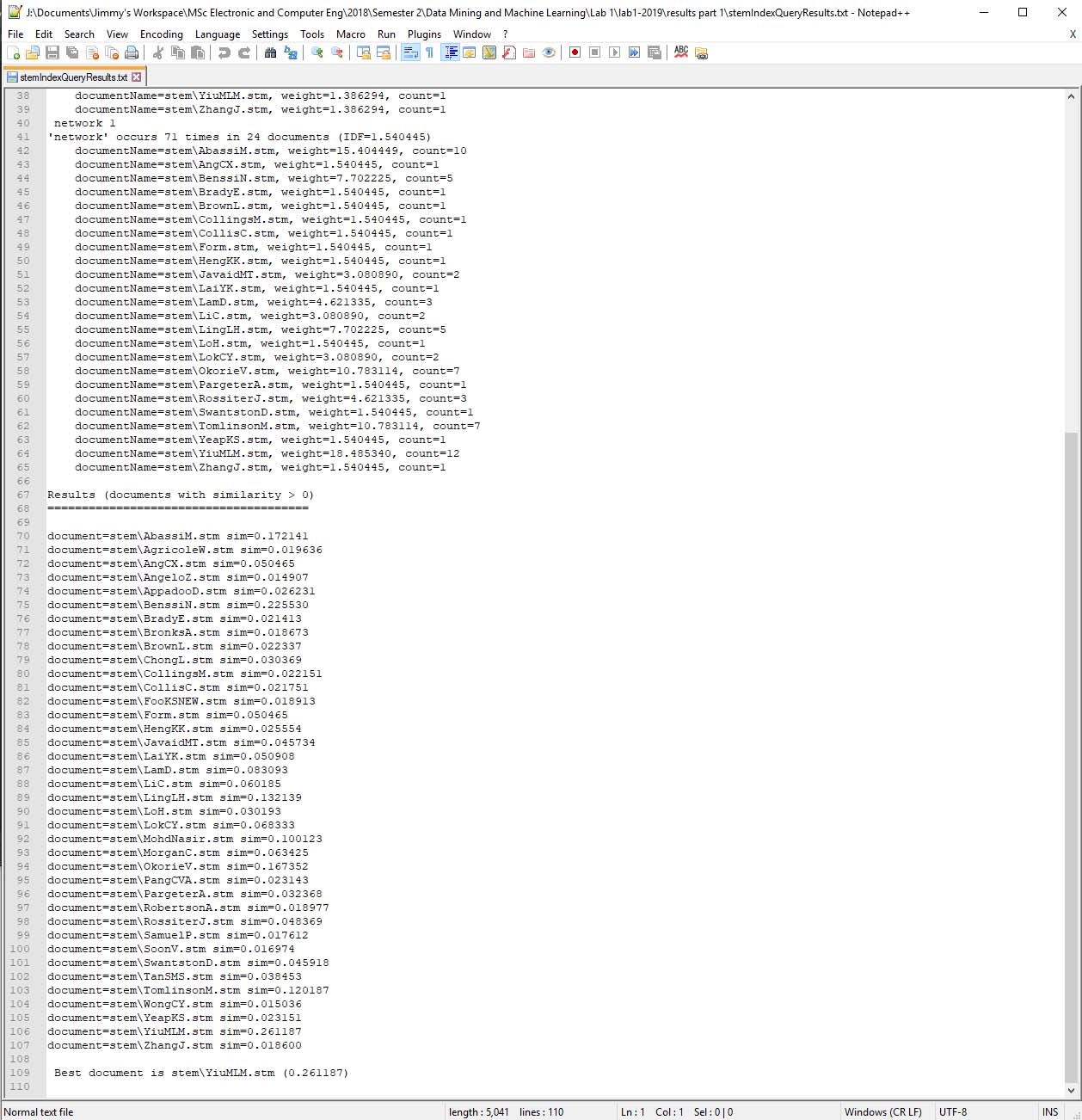


Figure 8: Retrieve on stemIndex file using query.stm (query.stp with stemming applied)

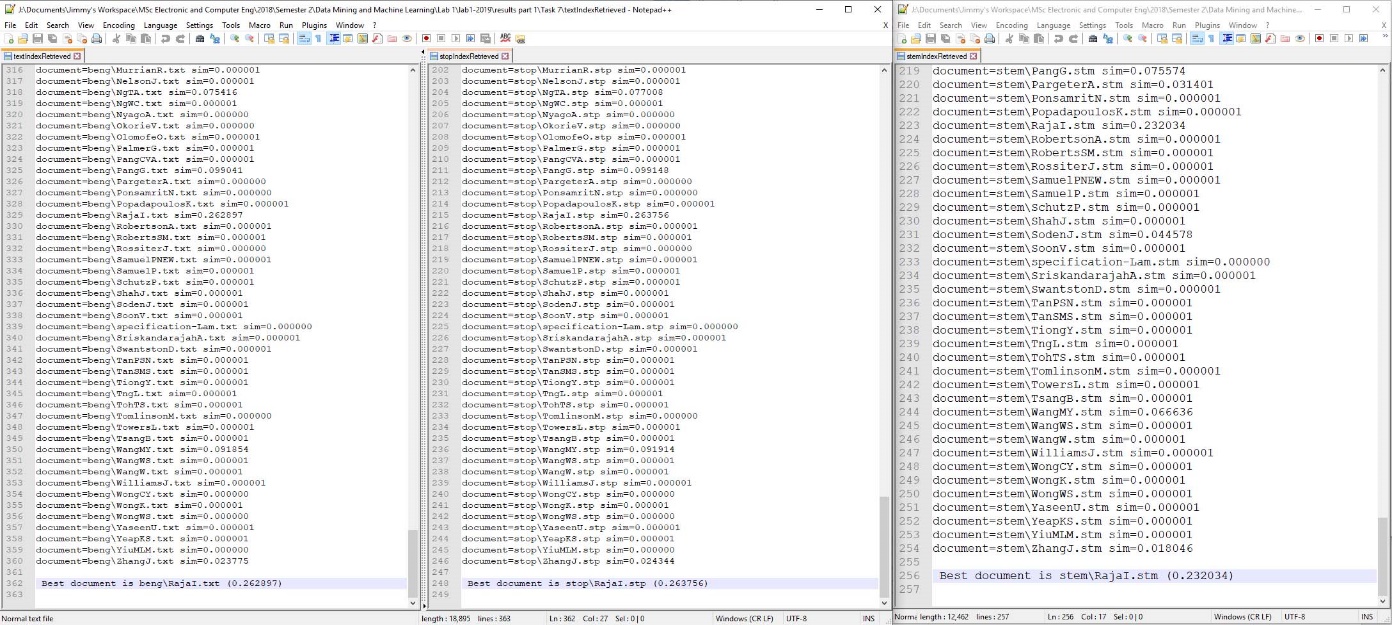


Figure 9: Retrieval Using Query, Query.stp and Query.stm (algorithm and adjacent)

### Questions

* + 1. Question 1:

AgricoleW.txt result list size = 374 words (in MS word)

AgricoleW.stp result list size = 242 words

374 – 242 = 132

132/374 \* 100 = 35.294117647058823529411764705882

Approx. = 35% reduction from using stop-word removal on AgricoleW.txt to AgricoleW.stp

* + 1. Question 2:

In the file AgricoleW.stm

“communications” stemmed to “commun”

“sophisticated” stemmed to “sophist”

“transmissions” stemmed to “transmiss”

* + 1. Question 3:

Document lengths of the documents: beng\DongP.txt, stop\DongP.stp and stem\DongP.stm

Table 1: Document lengths for DongP files

|  |  |
| --- | --- |
| Filename | Document length |
| beng\DongP.txt | 42.396210 |
| stop\DongP.stp | 42.392876 |
| stem\DongP.stm | 40.547955 |

The document lengths differed as stop-word removal and stemming had been applied. This was because some of the terms that contributed to the document length of DongP.txt had been reduced in DongP.stm.

The difference between DongP.stm and DongP.txt was greater than that between DongP.stp and DongP.txt because the stemming process applied to the documents reduced the terms significantly more than the 50 word stop-list. Stemming stripped morphological features of the terms and left behind fewer terms that would have been considered as independent otherwise.

* + 1. Question 4:

The IDF of the term “adjacent” was so close to zero because the term occured in almost all the documents except one. Since ND (the number of documents in the corpus) 112 was almost equal to NDt (the number of documents that include the term “adjacent”) 111, hence IDF (adjacent) was almost 0 (ND = NDt).

Table 2: Entries for the term "adjacent" (found in textIndex and StemIndex files)

|  |  |
| --- | --- |
| Filename | Word: “adjacent” |
| textIndex | 141 word=adjacent wordCount=118 docCount=111 IDF=0.008969 |
| stopIndex | 140 word=adjacent wordCount=118 docCount=111 IDF=0.008969 |

* + 1. Question 5

Table 3:: Entries for the term "algorithm" (found in textIndex, stopIndex and stemIndex files)

|  |  |
| --- | --- |
| Filename | Word: “algorithm” |
| textIndex | 185 word=algorithm wordCount=14 docCount=7 IDF=2.772589 |
| stopIndex | 184 word=algorithm wordCount=14 docCount=7 IDF=2.772589 |
| StemIndex | 152 word=algorithm wordCount=36 docCount=15 IDF=2.010449 |

The word ranking of the term “algorithm” had gone up from 185 to 184 and then 152 in the corpus of 112 documents (beng, stop and stem respectively). It could be assumed this was because some words had been removed as stopping and then stemming had been applied to the different folders.

The wordCount and docCount of “algorithm” in the stemmed documents went up in number. This was because the porter-stemmer reduced the variations such as “algorithms”, “algorithmic” into a simpler form of “algorithm” hence increasing its count within the stemIndex file.

### Conclusion

Information retrieval based on a query or search criteria from a large corpus of documents using conventional methods such as manually searching through the corpus would be time consuming and hard. However, the use of statistical calculations such as Term-frequency Inverse-document frequency (TF-IDF) gives an acceptable alternative to search through a large corpus as demonstrated in this lab.

In-order to have consistent results with the TF-IDF, Stop-word removal and Stemming are applied to the corpus of documents. Stop-word removal helps in removing noise words that usually would give the wrong information regarding a term’s re-occurrence within a document. Stemming, in this case porter-stemmer also helps in reducing the amount of terms that may be from an equivalent class (synonymous). This has been verified through the results obtained in this lab.

TF-IDF weighs the significance of a given term through calculations of how much the term occurs in a document as well as the number of documents it occurs in within the corpus. The results are stored within an index that is used as a future reference point for term queries (since the occurrence of those terms never change within the corpus if more documents are not added to it).

The lab results demonstrate how effective TF-IDF is in retrieving text from a large corpus of documents and how it can be implemented.

### References

Belew, R.K. (2000) **Finding Out About: A cognitive Perspective on Search Engine Technology and the WWW.**1st ed. Cambridge: Cambridge University Press.

Jancovic, P (2019) **Data Mining and Machine Learning** [Online]. Available from: <https://canvas.bham.ac.uk/courses/34771> [Accessed 30 January 2019].