Text Retrieval

William L. and Andrea T.
Data Mining and Machine Learning

Part 1: TF-IDF Text Retrieval

Stop Word Removal

Going into the lab1-2021 directory inside the command prompt, we ran the command `stop stoplist50 docOrig\AbassiM.txt` to display the result of using stop word removal on the document *AbassiM.txt*. Because there are a total of 112 file to convert from a normal *.txt* file to a *.stp* file, it is better to write a batch file to convert everything at the same time. Therefore, executing `stopScript.bat return` places all new *.stp* files into the *DocStop* directory. Doing this gives us an additional 112 *.stp* files.

Q1: What is the percentage reduction in the number of words in a document as a consequence of stop-word removal? Specifically, what is the reduction in the case of the file *AgricoleW.txt*?

A1: The percentage reduction in no. of words using stop-word removal would usually be around 20-30% of words. As for the reduction of words for the *AgricoleW.txt*, as seen from Fig. 1, the word count decreases from 405 to 303, a reduction of 102 words, which is 25.2%.

```
Enter source file path: docOrig/AgricoleW.txt

Total characters = 2551
Total words      = 405
Total lines      = 36
```

```
Enter source file path: docStop/AgricoleW.stp

Total characters = 2070
Total words      = 303
Total lines      = 51
```

Fig. 1. Output showing the word count of the AgricoleW.txt using a C program. (a) word count before stop removal. (b) word count after stop removal

Stemming

From *.stp* files we can create *.stm* files. And we can use a similar batch script to convert all 112 *.stp* files to *.stm* files using the command `stemScript.bat return`, storing all *.stm* files into *DocStem*.

Q2: Find the file *AgricoleW.stm*. What are the results of applying the porter-stemmer to the words communications, sophisticated and transmissions?

A2: Communications → commun, sophisticated → sophist, transmission → transmiss

Document Index Files

In the directory, we have the executable file *index.c* that we can use to create a list of files under the name *textFileList*, *stopFileList* and *stemFileList.* And the command to run them is:

```
index textFileList > textIndex
```

```
index stopFileList > stopIndex
```

```
index stemFileList > stemIndex
```

These index files store the list, weights and their appearances in each document of *.txt*, *.stp* and *.stm* files.

Q3.1: What are the document lengths of the documents: *docOrig\DongP.txt*, *docStop\DongP.stp* and *docStem\DongP.stm*? Why are they different?

A3.1: From the three Index Files:

DongP.txt: 42.396210 (from *TextIndex*)

DongP.stp: 42.392876 (from *StopIndex*)

DongP.stm: 40.547958 (from *StemIndex*)

The values can be decimal values because it is based on TF-IDF values instead of the number of documents which are integers. They are different in length because the documents lengths are dependent on the term weights and because some terms are combined in the *.stm* files, the overall weight has decreased, resulting in a smaller document length.

Q3.2: Why is the difference between the document lengths of *docStem\DongP.stm* and *docOrig\DongP.txt* greater than the difference between the document lengths of *docStop\DongP.stp* and *docOrig\DongP.txt*?

A3.2: The difference between the *.stm* and *.txt* is greater than *.stp* and *.txt* because more words or tokens were lost between the *.txt* and *.stp*.

Q4: The IDF of the term *adjacent* is 0.009. Why is it so close to zero?

A4: Because, out of the 112 documents, *adjacent* appears in 111 documents, so it has less weight, and is less useful. This can be seen in Fig. 2, which is taken from the index file.

```
141 word=adjacent wordCount=118 docCount=111 IDF=0.008969
```

Fig. 2. Information on the word adjacent taken from the .txt index file

Q5: Find the word *algorithm* in the three index files. Explain why the entries for this word are different in the three files.

A5: Fig. 3 shows that the word *algorithm* has different entries in the different index files. This is because *TextIndex* includes stop words that aren't removed. *StopIndex* is similar to *TextIndex* without the stop words while *StemIndex* might combine different forms of words into one, decreasing the index count and number of entries.

```
185 word=algorithm wordCount=14 docCount=7 IDF=2.772589
  1 docName=docOrig\EftekhariS.txt count=2 weight=5.545177
  2 docName=docOrig\LokCY.txt count=1 weight=2.772589
  3 docName=docOrig\NgTA.txt count=1 weight=2.772589
  4 docName=docOrig\PangG.txt count=2 weight=5.545177
  5 docName=docOrig\RajaI.txt count=5 weight=13.862944
  6 docName=docOrig\WangMY.txt count=2 weight=5.545177
  7 docName=docOrig\ZhangJ.txt count=1 weight=2.772589
```

(a)

```
184 word=algorithm wordCount=14 docCount=7 IDF=2.772589
  1 docName=docStop\EftekhariS.stp count=2 weight=5.545177
  2 docName=docStop\LokCY.stp count=1 weight=2.772589
  3 docName=docStop\NgTA.stp count=1 weight=2.772589
  4 docName=docStop\PangG.stp count=2 weight=5.545177
  5 docName=docStop\RajaI.stp count=5 weight=13.862944
  6 docName=docStop\WangMY.stp count=2 weight=5.545177
  7 docName=docStop\ZhangJ.stp count=1 weight=2.772589
```

(b)

```
152 word=algorithm wordCount=36 docCount=15 IDF=2.010449
  1 docName=docStem\AliR.stm count=2 weight=4.020897
  2 docName=docStem\BenHasineA.stm count=4 weight=8.041795
  3 docName=docStem\BradyE.stm count=2 weight=4.020897
  4 docName=docStem\BronksA.stm count=2 weight=4.020897
  5 docName=docStem\ChanWK.stm count=1 weight=2.010449
  6 docName=docStem\EftekhariS.stm count=5 weight=10.052243
  7 docName=docStem\LokCY.stm count=4 weight=8.041795
  8 docName=docStem\MohdNasir.stm count=2 weight=4.020897
  9 docName=docStem\NgTA.stm count=1 weight=2.010449
  10 docName=docStem\PangG.stm count=2 weight=4.020897
  11 docName=docStem\PargeterA.stm count=1 weight=2.010449
  12 docName=docStem\RajaI.stm count=6 weight=12.062693
  13 docName=docStem\SodenJ.stm count=1 weight=2.010449
  14 docName=docStem\WangMY.stm count=2 weight=4.020897
  15 docName=docStem\ZhangJ.stm count=1 weight=2.010449
```

(c)

Fig. 3. The word algorithm taken from the three index files. (a) from txt index. (b) from stp index. (c) from stm index.

Retrieval

The query file contains the text: *communication and networks*. Applying stopping and stemming with the commands: `stop stoplist50 query > query.stp` and `porter-stemmer query.stp > query.stm`. With the newly created *.stp* and *.stm* files from the query, we can use *retrieve.exe* to get a result for which documents are best for getting the query.

Q6: Compare the results of these two searches with the result for the original raw text files. What do you conclude?

A6: From Fig. 4, it is concluded that after stemming, other terms in other documents were stemmed into the stemmed version of 'communication and networks', therefore, another document different from the stop and original version was selected as the best document for the query. As for the stop result, it has similar values because only useless and 'noise words' were removed.

```
Best document is docOrig\TomlinsonM.txt (0.152037)
Best document is docStop\TomlinsonM.stp (0.152309)
Best document is docStem\YiuMLM.stm (0.261187)
```

Fig. 4. Results from the newly created retrieve documents

2 Additional Queries for Retrieval

Creating 2 additional queries:

q1: *computers and laptops*

q2: *is digital technology the best?*

Stopping and stemming both files:

`stop stoplist50 q1 > q1.stp` + `porter-stemmer q1.stp > q1.stm`

`stop stoplist50 q2 > q2.stp` + `porter-stemmer q2.stp > q2.stm`

And using retrieve:

`retrieve textIndex q1 > RetrieveOrg_q1` + `retrieve stopIndex q1.stp > RetrieveStop_q1` + `retrieve stemIndex q1.stm > RetrieveStem_q1`

`retrieve textIndex q2 > RetrieveOrg_q2` + `retrieve stopIndex q2.stp > RetrieveStop_q2` + `retrieve stemIndex q2.stm > RetrieveStem_q2`

```
Best document is docOrig\RobertsSM.txt (0.077057)
Best document is docStop\RobertsSM.stp (0.077285)
Best document is docStem\RossiterJ.stm (0.156027)
```

Fig. 5. Results for the query q1: computers and laptops

```
Best document is docOrig\NyagoA.txt (0.230118)
Best document is docStop\NyagoA.stp (0.176783)
Best document is docStem\NyagoA.stm (0.158520)
```

Fig. 6. Results for the query q2: is digital technology the best?

From Fig. 5 and Fig. 6, the results of *q1* are pretty similar to that of the previous retrieval task using the query *communication and networks*. But the results of retrieval using *q2* has larger differences between the original and stop files because *q2* has more stop words than there are in *q1*.

Latent Semantic Analysis

Word-Matrix Document

The executable doc2vex.exe creates matrix W that can be applied to the stemmed documents using the command `doc2vec stemFileList > WDM`. This creates a document vector for each document in the *DocStem* directory and stacks them to create the file *WDM*.

Applying SVD to WMD

In MATLAB, the commands:

`W = load('WDM');` (reads the data in *WMD* into the MATLAB matrix W)

`[U,S,V]=svd(W);` (runs SVD on W, decomposing it as $W = USV^T$.)

Q1: Are the matrices U and V as you would expect? Explain.

A1: Yes, the matrices are as expected because U and V satisfies $\mathrm{UU^T = I = U^TU, VV^T = I = V^TV}$. Additionally, S is also as expected, and can be verified as correct by seeing, as shown in Fig. 7 that it satisfies

112x2743 double

	1	2	3	4	5	6
1	274.5965	0	0	0	0	0
2	0	53.7182	0	0	0	0
3	0	0	47.1338	0	0	0
4	0	0	0	44.0433	0	0
5	0	0	0	0	37.6685	0
6	0	0	0	0	0	33.4974
7	0	0	0	0	0	0
8	0	0	0	0	0	0

Fig. 7. Result of matrix S taken from MATLAB

Q2: What are the values of the first 3 diagonal entries in S?

A2: From Fig. 7, the first three diagonal entries are 274.596487768020, 53.7182065440782 and 47.1338164059137. These are singular vectors, or 'latent semantic classes', that corresponds to the columns of V.

Getting the 1st, 2nd and 3rd column of V in MATLAB:

```
sV1 = V(:,1); sV2 = V(:,2); sV3 = V(:,3);
```

The most important words that determine the interpretation of vector 1 are the biggest values (positive or negative). So, we also want to know the position/index of the biggest values, so we know which word it corresponds to. Fig. 8 shows that from vector 1, the biggest numbers are entries 1902, 1723, and 2325. These entries can then be linked back to the index files to see which words are the most significant for each singular vectors. In this case, we used *mink* instead of *maxk* in the MATLAB script because MATLAB was thinking that the max was the most negative and the min is the least negative.

```
[m1,am1] = mink(sV1,3)

m1 =
    -0.6151
    -0.2492
    -0.2451

am1 =
    1902
    1723
    2325
```

```
[m2,am2] = mink(sV2,3)

m2 =
    -0.1535
    -0.1191
    -0.1090

am2 =
    608
    2043
    576
```

```
[m3,am3] = mink(sV3,3)

m3 =
    -0.1708
    -0.0911
    -0.0895

am3 =
    2268
    608
    2181
```

Fig. 8. Results of applying mink/maxk to the vector to retrieve the index and maximum value in the singular vector

Q3: Find the three most significant words for each of the singular vectors sv1, sv2 and sv3. What is your interpretation of the corresponding semantic classes?

A3: The three most significant words from each singular vector are

sV1: project, outcom, student

sV2: data, reson, coupl

sV3: speech, data, should

From these significant words, we can interpret that sV1 was most likely about project specifications and introductions, sV2 was about results and analysis, and sV3 about conclusions.