Homework 2 Report

Paul Sampson Ledala

pledala1@umbc.edu,

Department of Computer Science and Electrical Engineering

University of Maryland, Baltimore County

Date: 3/21/22

I. Introduction

Term Weighting is the process of assigning weights to the terms in the document based on the frequency of the token in the given document and it's rarity in the corpus of data. In this homework we were supposed to extend the preprocessing and calculate the term weights of the tokens. The input files were arranged in a directory called 'files'. These were HTML files which needed prior conversion by pre-processing to give tokens. Then the preprocessed documents were to be term weighted one by one. Finally, the indexed time is to be plotted on the graph. The following report explains how the preprocessing is extended from the earlier assignment, term-weights calculated by the tf-idf formula and indexing time appears in the program.

II. Specifications of the Program

Input: Directory of HTML files named 'files', stoplist.txt

Output: Directory of term-weighted tokens, graph of indexing time function

Example command to run the code including input and output paths:

python calcwts.py

C:\Users\p\Documents\Coding\Github\IR_Tokenization\Information_Retrieval\files

C:\Users\p\Documents\Coding\Github\IR_Tokenization\Information_Retrieval\output

(includes a space between calcwts.py and the input path, and a space between input and

output path)

Coding Language used: Python 3.7.1

Parser used: BeautifulSoup

Tokenizer used: wordpunkt tokenizer of the nltk package

Term Weights formula: tf * idf

III. Methods Used in the Program:

A. Extending Preprocessing

In the earlier HW, both punctuation and numbers were removed from the list of tokens while writing into the file. As the wordpunct tokenizer was used, this separated the punctuations into a separate token. This was included in the tokens list but eradicated before copying the tokens into the output files. In this homework, this functionality was extended by removing the stopwords from the tokens obtained. The list of tokens which were of length 1 were also removed. The tokens whose frequency was 1 in the entire corpus were also removed as they could be one-off terms which don't mean anything.

B. Term Weighting formula:

The term weights are calculated using the tf-idf formula. Here tf is the term frequently of the token in the document and the idf is the inverse document frequency of the document in the corpus.

Term Weight = tf * idf = (tf(w,i)/|Di|) * log(|C|/df(w))

Here, tf (w,i): term frequency of the token w in the document i
|Di|: no. of tokens in the document i
|C|: no. of documents in the corpus
df (w): no. of documents containing the token w

These term weights that are calculated are stored in a list of dictionaries called all_docs_tokens. This contains files of terms and their weights. The Inverted Indexes of each term are calculated, along with their frequency (docs_containing_token), and stored in a dictionary. So, this is a memory-based algorithm.

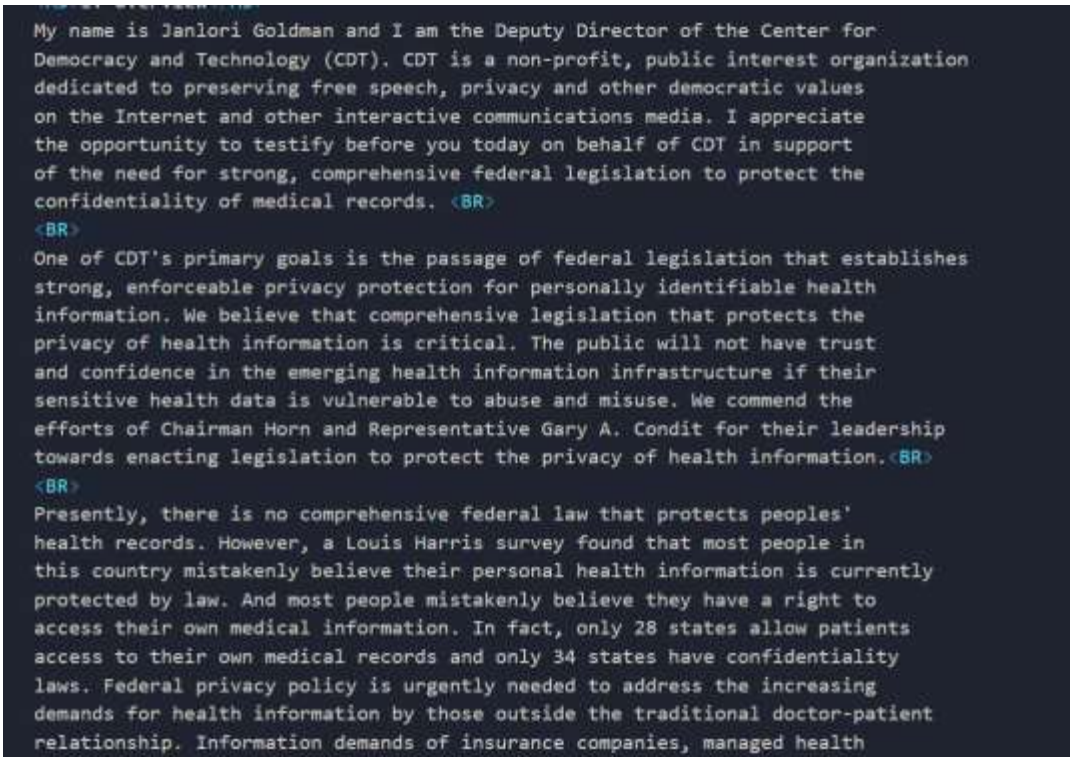
C. Input Documents and Term Weights:

The following screenshots show couple of examples of the input html files and the resulting term weights calculated for all the preprocessed tokens

```
</B><hr><BR>
In recognition of more than two decades of courageous journalism, in defiance of political pressure and violence that

When Blancornelas and longtime colleague Hector F&eacute;lix Miranda co-founded <I>Zeta</I>, a feisty weekly newspape

The cost of <I>Zeta&#146;s </I>independence has been high: F&eacute;lix, a popular columnist known as "F&eacute;lix t

F&eacute;lix&#146;s death is one of 10 cases documented by CPJ over the past decade of Mexican journalists who were m

<I>Zeta's</I>  drama has played out in one of the most contentious and volatile regions of Mexico. In 1989 Baja Calif

In recent years Tijuana has been one of the bloodiest battlegrounds in Mexico&#146;s ongoing internecine drug war whi

Despite the risks, Blancornelas has not been deterred. Largely inspired by his example, a newly empowered generation

In the spirit of the International Press Freedom Awards, whose previous recipients have included the courageous Irish
<B><BR>
THE UPHILL CLIMB TO ESTABLISH AN INDEPENDENT VOICE IN TIJUANA<BR>
</B><BR>
Blancornelas and F&eacute;lix launched <I>Zeta</I> soon after losing control of another Tijuana newspaper, <I>ABC</I>

Fearful that <I>Zeta</I>, which picked up where <I>ABC</I> left off, would be a target of government reprisal, Blanco

<B>FELIX MURDER ENRAGES MEXICAN PUBLIC</B><P>

In 1988, F&eacute;lix, who served as co-editor of <I>Zeta</I>, was assassinated while driving to work along a narrow
```

```
angry 0.09904590555071512
anh?nger 0.033062640226720036
animal 0.24163862416441814
animals 0.19931676309782678
animation 0.04595205817144385
anl 0.3244238095489049
ann 0.15748014508922073
annak 0.5446126545959222
annal 0.17210691863539396
anne 0.09904590555071512
annex 0.2736539446389372
annie 0.057824116614398825
anniversary 0.033062640226720036
annotated 0.0689396734429355
announced 0.3446983672146775
announcement 0.0794523705843258
announcements 0.057824116614398825
announces 0.04595205817144385
annual 0.37407657214609225
annulment 0.0689396734429355
annyat 0.033062640226720036
annyi 0.1791504614654745
annyira 0.2181955162688081
annyit 0.1791504614654745
ann?lis 0.033062640226720036
anoka 0.033062640226720036
anomalies 0.0689396734429355
anomaly 0.057824116614398825
anonymous 0.10825284937058009
anorexia 0.09904590555071512
```

```
aa 0.06355773862710902
aaa 0.05265701042903317
aachen 0.01917419925951215
aar 0.016637175527413627
aaron 0.02612460382209086
aarp 0.021590235293040574
aascpa 0.007978989765696887
aau 0.046515546602141626
ab 0.1270841696159685
ababa 0.007978989765696887
abacom 0.023902696859782403
aban 0.007978989765696887
abandon 0.007978989765696887
abankoknak 0.007978989765696887
abastecimiento 0.007978989765696887
abated 0.007978989765696887
abb 0.06960732433076937
abbahagyta 0.007978989765696887
abban 0.11099095654148457
abbo1 0.03979213151655966
abc 0.056941346297169286
abdn 0.007978989765696887
aber 0.032338451866770444
aberdeen 0.032338451866770444
abetting 0.023902696859782403
abiertos 0.01395466398064249
abilene 0.01917419925951215
abilities 0.04153447313888843
ability 0.11302514471821606
able 0.10532345334570696
```



D. Efficiency as a function of documents indexed:

In preprocessing, the program's runtime complexity depends on the following methods as they are inside

the loop which iterates through all the input files:

1. BeautifulSoup
2. wordpunct_tokenize
3. Counter

BeautifulSoup's HTML parser uses a DOM tree based approach where traversing the tree is the most complex action which takes O(n) time complexity (where n is the number of tags in HTML). Constructing a Counter (among all the dictionary operations) takes the maximum runtime complexity which is O(n) (where n is the number of keys inserted). Wordpunct tokenizer is a regular expression based tokenizer which takes O(n) (where n is the number of characters in the input) Using this information, we can calculate that the most time consuming part is the tokenizer which takes O(m) where m is the number of characters in the input file of the tokenizer. This O(m) is the runtime complexity per input file. If there are n input files in the program, and the largest file has m characters, then the runtime time complexity can be approximated to O(nm).

Worst Case Time Complexity = O(nm), where

n is the number of input html files

m is the number of characters in the largest input html file

We see even in calcwts.py that there are only two loops one over the input files n and the other over tokens of each file m (in lines 32 and 34). The heavy lifting is done by the dictionaries which store:

- Tokens of all the documents separated by document
- Documents containing a particular key token

Apart from this the length of each document and the length of the whole corpus are also stored. So the time complexity remains O(mn) even for the term weighting. This can be seen by the indexed time function plotted by the program which shows documents processed vs time taken. Total CPU time elapsed in secs = 318 (on an I5 RAM).