BUILDING A MINI SEARCH ENGINE

Preprocessing Phase

By

Rajendra Banjade

The University of Memphis, TN, USA

2012

A Perl program that preprocesses a collection of documents. The input to the program will be a directory containing a list of text files. The documents must have been converted to text before using them.

Removes the following during the preprocessing:

- digits

- punctuation

- stop words (given in english.stopwords.txt)

- urls and other html-like strings

- uppercases

- Morphological variations

Source code:

```
# Does some preprocessing on different web page contents crawled and saved in
a folder. Removes the following during the preprocessing:
#    - digits
#    - punctuation
#    - stop words (use the generic list available along with this code,
#english.stopwords.txt)
#    - urls and other html-like strings
#    - uppercases
#    - morphological variations
#
# VERSION HISTORY:
# Rajendra Banjade 10/5/2012
#  The University of Memphis, TN, USA
#
#
require "stemmer.pl";                                # porter stemmer
(http://tartarus.org/~martin/PorterStemmer/)

my $preprocessedPageCount = 0;                       # counter for preprocessed
pages.
my $tokenCount = 0;                                  # total number of tokens
found
my $wordCount = 0;                                   # total number of words
remained (after removing stop words etc).

# stop words hash
%stopWords = {};

# initialize the porter stemmer
&initialise();
```

```
# module to trim any whitespace in the string.
sub trim($)
{
    my $s = shift;
    $s =~ s/^\s+//;
    $s =~ s/\s+$//;
    return $s;
}

# Populate stop word hash with the list of words from a file.
open(SWFILE, "<stopwords.txt" )  or die("Failed to read stop word file:
stopwords.txt\n");
while (<SWFILE>) {
    $stopWords{&trim($_)} = 1;
}
close (SWFILE);

# For each raw file, remove links and html tags, stopwords, punctuation etc.
And save to a file (if anything remains).
opendir(RAWDIR, "raw") || die "Can't opedir: $!\n";
while (readdir(RAWDIR)) {
    $file = ($_);
    next unless (!($file =~ /^\./));              # skip file that starts
with ., some hidden files.
    open (RAWFILE, "<raw/$file") || die ("Failed to ope :raw/$file");

    # open a file in the preprocessed folder for each raw file and write the
preprocessed texts.
    open (OUTFILE, ">preprocessed/$file") || die ("Failed to create output
file:  $preprocessed/$file");

    $fileContent = join (" ", <RAWFILE>);       # bring whole content in a
single string. Processing linewise can be difficult while applying regular
expressin.

    $fileContent =~ s/\s+/ /g;                  # replace any sequence of
whitespaces by a single whitespace (for easy split)
    $fileContent =~ s/<.+?>//g;                    # remove valid html tags
(opening | closing), do shortest match (?).
    $fileContent =~ s/[0-9]+//g;                   # remove any digits
    $fileContent =~ tr/[A-Z]/[a-z]/;               # change everything to
lowercase
    $fileContent =~ s/ //g;                   # remove nbsp;
    $fileContent =~ s/(http:\/\/|ftp:\/\/)?(\.?\w+-
*\w+)+\.(com|net|org|gov|edu)\/*//gi; # remove urls in text, <taken from
web>.
    $fileContent =~ s/mailto://g;                  # remove mailto:
    $fileContent =~ s/(.+)@(.+)\.(.{2,4})?//g;     # simple regex to remove
some email addresses, maynot match all email addresses.
    $fileContent =~ s/[[:punct:]]/ /g;          # remove puncutation
    $fileContent =~ s/[|||]?//g;                # remove quotes, they are
still there.
    my @tokens = split /\s+/, $fileContent;     # split by whitespace(s)
    $tokenCount += @tokens;                     # these are the tokens (before
removing stop words)

    my @words = ();                             # array that contains index
```

```
words.
    # foreach token, do not add to index words if it is an stopword.
    foreach $word (@tokens) {                  # Case sensitive, match.
        if (! (exists $stopWords{$word})){
            $word = &stem ($word);
            #$word =~ s/'//g;                  # remove apostrophe
            if($word){
                push @words, $word;
            }
        }
    }
    if (@words >0) {
        print OUTFILE join(" ", @words ) . "\n";    # write line to file
    }

    $preprocessedPageCount++;
    close (RAWFILE);
    close (OUTFILE);
}
closedir(RAWDIR);

# print some statistics
print ("\n\n Total files processed: $preprocessedPageCount , Total word
count: $tokenCount. \n\n");
```

Src/Preprocessor.pl

NOTE:

- Perl provides a concise and robust form to remove punctuations (by doing `$fileContent =~ s/[[:punct:]]/ /g`). It removes the apostrophe (‘) as well. Stemming after removing punctuation prevents removing stop words that contain apostrophe. I noticed that problem at the last moment and it is still there.

To run the program,

- Go to the <your directory>/src
- Issue command, perl Preprocessor.pl

The program,

1. Reads each raw file from the folder *raw.* It contains the raw files fetched during the assignment #3 and other 10 news texts from news.yahoo.com.
2. Removes html tags.
3. Converts to lowercase and removes digits, punctuation, stop words, urls and email addresses etc.

4. Write the remaining words (index words) to a file in the folder *preprocessed* with the same name.

Sample output:

The preprocessed content for the information retrieval home page content is,

```
inform retriev institut intellig system univers memphi inform retriev web
search prof vasil ru comp psyc fall announc inform schedul announc midterm
date oct assign issu perl slide phd student submit titl short descript
present sept bookmark page vru teach ir websearch inform retriev web search
inform retriev web search address major problem time todai major problem lack
inform inform intellig wai organ vast amount inform fingertip effect search
face inform overload problem inform retriev web search class present major
challeng pose problem solut challeng introduc comput techniqu search inform
static collect document dynam collect web student expos text process
algorithm classic inform retriev model boolean vectori model web search
techniqu close relat natur languag process catalog entri advanc current
research topic databas inform manag emphasi nontradit data applic prerequisit
comp permiss instructor lectur pm fit prof vasil ru ta tbd ta vhaduri gmail
dot offic hr prof appt fitc ta tbd web page vru teach ir websearch textbook
baeza yate ribeiro neto modern inform retriev requir man raghavan schutz
introduct inform retriev recommend polici grade assign project midterm final
submiss lectur late lectur plagiar lectur intellectu engag lectur announc
class read page similarir cours slide adapt cours assign gener grade correct
robust qualiti solut document style assign written code requir code shoud
follow recommend standard code style standard recommend plagiar cheat polici
plagiar cheat behavior form uneth detriment proper educ toler work submit
student project program assign lab assign quizz test expect student work
plagiar incur part work pass proper credit list sourc work reader led effort
student allow encourag discuss resourc literatur includ internet assign refer
includ materi consult citat made materi verbatim plagiar cheat occur student
receiv fail grade assign instructor discret fail grade instructor decid
forward incid univers judici affair offic disciplinari action inform code
student conduct academ disciplin procedur refer jaffair tent schedul find
lectur assign lectur compil sourc includ person note textbook materi relat
class inform retriev natur languag process famou univers primarili ut austin
dr rai moonei unt dr rada mihalcea materi week mondai wednesdai read amp
assign introduct ppt introduct ppt chapter assign introduct perl ppt
introduct perl ppt perl tutori introduct perl ppt ir model ppt assign retriev
evalu ppt retriev evalu ppt chapter assign queri languag ppt queri oper ppt
chapter queri oper ppt text properti ppt chapter text oper ppt index search
ppt chapter porter stemmer porter stemmer slide assign class fall break web
crawl review web search intro ppt midterm chapter assign web search intro ppt
web search ppt assign pagerank paper chapter text categor ppt text categor
ppt text cluster ppt text cluster ppt chapter assign question answer ppt
```