**Supplemental File 2**

**Title:**

A snapshot of 3,649 web-based services published between 1994 and 2017 shows a decrease in availability after two years

**Authors:**

Ágnes Ősz, Lőrinc Sándor Pongor, Danuta Szirmai, Balázs Győrffy

**Description:**

Parses Pubmed abstract file, and outputs

#basic info of every article such as PMID, PMCID, journal name, publication year, Title, DOI, URL

**Usage:**

perl parser.pl <input PUBMED abstract file>

**Requirements:**

Parser input file

**Script:**

====================================================================

#DOI number, issue of journal, and url if present in abstract

use strict; use warnings;

#Stores current Pubmed article entry read

my @entry;

#Print header of output table

print "Journal\tYear\tPMID\tPMCID\tDOI\tissue\ttitle\turl\ttext\n";

#read pubmed abstract file, line-by-line with while() cycle

open FH, $ARGV[0] or die; #open file

while(<FH>) { #iterate through file

 chomp($\_); #remove whitespace

 if($\_ =~ m/(^\d+|^\d)(\.\ )(.+)/ && index($\_, "©") == -1) { #check if line is new entry

 if($#entry > 1) { #check if number of lines from previous entry is more than 1

 ParseEntry(@entry); #parse previous entry

 }

 undef(@entry); #empty entry array

 }

 push @entry, $\_; #add new entry line to array

}

close(FH); #close file

#parser sub-routine: Parses entry read in main cycle

sub ParseEntry {

 my @data = @\_; #get input array from @\_ variable

 my $journal; #stores journal name

 my $title; #stores article title

 my $year; #stores publication year

 my $issue; #stores journal issue

 my $pmid; #stores PUBMED ID

 my $pmcid; #stores PMC ID

 my @https; #stores http address from abstract

 my @wwws; #stores stores www address from abstract

 my $doi; #stores DOI number of article

 my $abstract; #stores abstract, for address extraction

 #parses header section of entry

 if($data[0] =~ m/(^\d+|^\d)(\.\ )(.+)/ && index($\_, "©") == -1) {

 my @tok = split /\./, $3;

 my @additional = split /\;/, $tok[1];

 my @sp = split " ", $additional[0];

 $journal = $tok[0]; #journal name

 $year = $additional[0]; #publication year

 $year =~ s/[a-zA-Z]+//g;

 $year =~ s/ //g;

 $year = substr($year, 0, 4);

 $issue = $additional[1]; #journal issue

 }

 my $linecount = 0;

 #extracts abstract from entry

 for(my $i = 0; $i <= $#data; $i++) {

 $data[$i] =~ s/\"//g;

 if($linecount == 1) {

 if(!defined $title) {

 $title = $data[$i];

 }

 else {

 $title = $title." ".$data[$i];

 }

 }

 if($linecount == 4) {

 if(!defined $abstract) {

 $abstract = $data[$i];

 }

 else {

 $abstract = $abstract." ".$data[$i];

 }

 }

 if(length($data[$i]) < 1) {

 $linecount++;

 }

 }

 @https = ReturnHttp($abstract); #checks for http address, and returns value

 @wwws = ReturnWww($abstract); #checks for www address, and returns value

 push(@https, @wwws); #concatenates the arrays

 my $longest = 0;

 my $long\_index = 0;

 #extracts longest http or www address

 for(my $i = 0; $i < $#https; $i = $i + 2) {

 if(length($https[$i]) > $longest) {

 $long\_index = $i;

 $longest = length($https[$i]);

 }

 }

 #Extracts PMID, PMCID and DOI numbers using regexp

 for(my $i = 3; $i <= $#data; $i++) {

 if($data[$i] =~ m/(^PMID\: )(\d+)(.+)/) {

 $pmid = $2;

 }

 elsif($data[$i] =~ m/(^PMCID\: )(.+)/) {

 my @temp = split " ", $2;

 $pmcid = $temp[0];

 }

 elsif($data[$i] =~ m/(^DOI\: )(.+)/) {

 $doi = $2;

 }

 }

 while($https[$long\_index] =~ m/(.+)([\.\,\)]$)/) {

 $https[$long\_index] = substr($https[$long\_index], 0, length($https[$long\_index]) - 1);

 }

 #print extracted data, if found (or defined)

 if(defined $journal) {

 print "$journal"; #print journal name

 }

 print "\t";

 if(defined $year) {

 print "$year"; #print publication year

 }

 print "\t";

 if(defined $pmid) {

 print "$pmid"; #print pubmed id

 }

 print "\t";

 if(defined $pmcid) {

 print "$pmcid"; #print pmc id

 }

 print "\t";

 if(defined $doi) {

 print "$doi"; #print doi number

 }

 print "\t";

 if(defined $issue) {

 print "$issue"; #print journal issue

 }

 print "\t";

 if(defined $title) {

 print $title; #print article title

 }

 print "\t";

 if(defined $https[$long\_index]) {

 print "$https[$long\_index]"; #print http address

 }

 print "\t";

 if(defined $https[$long\_index + 1]) {

 print "$https[$long\_index + 1]"; #print www address

 }

 print "\n";

}

#returns fraction of abstract starting with http, and separated by space

sub ReturnHttp {

 my $in = $\_[0];

 my @res;

 my $j = 0;

 while(index($in, "http") != -1) {

 $in = substr($in, index($in, "http"));

 my @tok = split " ", $in;

 my $short = substr($in, 0, length($tok[0]) + 10);

 push @res, $tok[0];

 push @res, $short;

 $in = substr($in, 1);

 $j++;

 }

 return @res;

}

#returns fraction of abstract starting with http, and separated by space

sub ReturnWww {

 my $in = $\_[0];

 my @res;

 my $j = 0;

 while(index($in, "www") != -1) {

 $in = substr($in, index($in, "www"));

 my @tok = split " ", $in;

 my $short = substr($in, 0, length($tok[0]) + 10);

 push @res, $tok[0];

 push @res, $short;

 $in = substr($in, 1);

 $j++;

 }

 return @res;

}