A simple Perl tokenizer and stemmer for biomedical text

Vetle I. Torvik, Neil R. Smalheiser, Marc Weeber
Department of Psychiatry MC912
University of Illinois at Chicago
1601 W Taylor Street, Chicago, IL 60612
vtorvik@uic.edu; neils@uic.edu; marc@weeber.net

Abstract

We present a simple Perl script tokenizer and stemmer which we developed in order to process biomedical text so that the resulting words and phrases remain meaningful and can be linked to slight variants in other titles.

Background

A variety of stemming algorithms, designed to strip text words to their stem, base or root forms, have been described and evaluated in the literature [1]. The Porter algorithm [2, 3] is by far the most popular, and although a variety of alternative stemmers have been made available [4], these were designed with the intention of optimizing retrieval of general purpose English documents. To our knowledge, no stemmer or tokenizer has been developed specifically for biomedical text. Here we present a simple open source Perl script tokenizer and stemmer (fig. 1) which we originally developed in order to modify the title of a biomedical journal article so that the resulting words and phrases remain meaningful and can be linked to slight variants in other titles.

Implementation

The biomedical stemmer employs some of the same rules as Porter (e.g., plural –s stemming) but contains a larger set of exceptions (fig. 1). In general, our script is more conservative than Porter - we do not try to strip all words to their basic stem, as that would give many words and phrases different meanings or could create artificial concepts (e.g., "phosphorylation of frizzled" would be stemmed as "phosphoryl of frizzl"). The script uses simple regular expressions and is self-contained, i.e., it does not use a dictionary and does not attempt to employ sophisticated algorithms such as substring matching [5].

Unpublished technical report. May, 2007.

In general, the biomedical stemmer leaves unchanged many terms that are (inappropriately) altered by the Porter stemmer, for example:

Title of a MEDLINE article:
Systemic oncolytic herpes virus therapy of poorly immunogenic prostate cancer metastatic to lung.

Porter stemmer output:
System oncolyt herp viru therapi of poorli immunogen prostat cancer metastat to lung.

Biomedical stemmer output:
Systemic oncolytic herpes virus therapy of poorly immunogenic prostate cancer metastatic to lung.

Conversely, the biomedical stemmer contains a few extra rules that take into account British (e.g., “tumour” -> “tumor”) and Latin spelling variations (e.g., “larvae” -> “larva”, “oestrogen” -> “estrogen”). For example:

Title of a MEDLINE article:
The carbohydrate components of the vagina of the normal and ovariectomized mouse during oestrogenic stimulation.

Porter stemmer output:
The carbohydr compon of the vagina of the normal and ovariectom mous dure oestrogen stimulation.

Biomedical stemmer output:
The carbohydrate component of the vagina of the normal and ovariectomized mouse during estrogenic stimulation.

The tokenizer carries out a number of basic operations such as making all text lower case, replacing parentheses and punctuation by spaces, remove initial and trailing spaces, etc. (fig. 1). The tokenizer also removes many non-alphanumeric characters -- which, it should be noted, may alter specific characteristics of chemical names (e.g., “(3-aminopropyl)-1-(3-pyridyl)-1,2,3,4-tetrahydro-beta-carbolines” is processed to “3-aminopropyl 3-pyridyl 4-tetrahydro beta carboline”). These are not common in our sphere of application but should be kept in mind by potential users who deal with text that is rich in chemical formulas.

Results and Discussion

The biomedical tokenizer/stemmer has been employed to process all of the titles in Medline (used to create databases in support of the Arrowsmith two node search interface [6]), and we have not encountered any scalability issues; as we have encountered new exceptions through use, we have made minor additions to the code. Thus, the stemmer and/or tokenizer should be useful for processing English biomedical sentences in general.

Conclusions

A free, public web interface has been implemented [7] which takes text (plain or rich format) as input. The user can specify whether to employ a sentence splitter on the input text (or not), to employ our tokenizer (or not), and to employ either the Porter stemmer or the biomedical stemmer. In addition, the user can specify either a short PubMed stoplist of 365 common words or no stoplisting at all. The output can be viewed in the web browser or exported to a text file. We invite comments from the user community so that the interface can be adapted further to a wide variety of needs.

Availability and requirements

- **Project name:** Biomedical tokenizer and stemmer
- **Project home page:** http://arrowsmith.psych.uic.edu
- **Programming language:** n/a
- **Other requirements:** web browser.
- **License:** ***.
- **Any restrictions to use by non-academics:** none.

Acknowledgements

This Human Brain Project/Neuroinformatics research is funded jointly by the National Library of Medicine and the National Institute of Mental Health.

References

[1] **Stemming.** [http://en.wikipedia.org/wiki/Stemmer].
[2] M.F. Porter, An algorithm for suffix stripping, Program 14(3) (1980) 130−137.
[3] **The Porter stemming algorithm** (revised January 2006). [http://www.tartarus.org/martin/PorterStemmer/].
[4] **The Lancaster stemming algorithm**.

[http://www.comp.lancs.ac.uk/computing/research/stemming/index.htm].
[5] B. Han, Z. Obradovic, Z.Z.Hu, C.H. Wu and S. Vucetic, Substring selection for biomedical document classification, Bioinformatics 22(17) (2006) 2136-2142.
[6] N.R. Smalheiser, V.I. Torvik, A. Bischoff-Grethe, L.B. Burhans, M. Gabriel, R. Homayouni, A. Kashef, M.E. Martone, G.A. Perkins, D.L. Price, A.C. Talk and R. West, Collaborative development of the Arrowsmith two node search interface designed for laboratory investigators, Journal of Biomedical Discovery and Collaboration 1 (2006) 8.
7.**Arrowsmith: Linking documents, disciplines, investigators and databases.** [http://arrowsmith.psych.uic.edu**].**

Figure 1. Perl scripts for tokenizing and stemming biomedical text.

```
sub tokenize {
    #input a string of words
    my $ti = shift;

    # add initial and final space
    $ti = " $ti ";
    # optimize regexp with study (12% speed increase)
    study $ti;
    # 1: make it all lowercase
    $ti = lc $ti;
    # 2: replace parentheses and punctuation by spaces
    $ti =~ s/[\[\](),\.\";:!?&\^\/\*]/ /g;
    # 3: remove --
    $ti =~ s/--+/ /g;
    # 4: hyphenation: replace by " " only before non-digit + ' and after non-digit
    $ti =~ s/([^\d\'])-([^\d])/$1 $2/g;
    # 5a: remove 's
    $ti =~ s/\'s / /g;
    # 5b:  ' : replace by " " only after non-digit
    $ti =~ s/([^\d])\'/$1 /g;
    # 6: remove words without any alpahabetical thingie (numbers, 100% +/-, etc)
    $ti =~ s/ [^a-z]+ / /g;
    # 7: some details
    $ti =~ s/ [^ ]+-(year|yr|month|week|day|hour|second) / /g;
    # umpieth
    $ti =~ s/ \d+th / /g;
    # cleanup:
    #8: remove non-alphanumerics in beginning and end (lots of chemical names that
were split)
    #     deleterious to short things ending with + (e.g. na2+ k+)
    $ti =~ s/ ([^a-z0-9]+)/ /g;
    $ti =~ s/[^a-z0-9]+ / /g;
    # 9: remove more than one space
    $ti =~ s/  +/ /g;
    # 10: remove initial and trailing spaces
    $ti =~ s/^ +//;
    $ti =~ s/ +$//;

    return $ti;
}


sub stem {
    #input a word
    my $w = shift;

    #British spelling -our
    $w =~ s/(tum|vap|odo|lab|behavi|flav|harb|fav|hon|col|rum|vig)our/$1or/g;
    #some Latin
    #  convert plural -ae to e: minimal matching using ? more efficient?
    $w =~ s/([a-z]{3,}?a)e$/$1/g;
    #  convert oe- to e- except oedipus (would be nice to replace all oe's ecept coe
?)
```

```
    $w =~ s/^o(e[a-z]{3,}?)/$1/g and $w =~ s/^(edipus|edipal)/^o$1/g;
    #consider chopping -s on words consisting entirely of alphabetical characters
    if ($w =~ /^[a-z]+s$/) {
  if (length( $w ) < 5) {
      $w =~
s/^(rat|dog|cat|ray|egg|cow|eye|age|use|boy|jaw|dye|fat|way|law|hen|leg|gel|bed|bat|ea
r|rod|sow|arm|pig|tip|end|ion|oil|toe|lip|day|fee|hit|gum|aim|sin|hog|act|eeg|jew|map|
job|rib|war|art|kit|oat|see|sea|bee|put|spa|toy|add|bud|pen|net|set|cue|get|jar|fit|ba
g|pup|one|pad|bar|car|cut|dot|lie|ant|bid|bin|hip|key|joy|lab|lid|nun|mri|pie|die|ewe|
eel)s$/$1/;
  }
  else{
      #don't mess with -ss, -us, -is, -phos
      $w =~ /(ss|us|is|phos)$/ or
      #some special -es
      $w =~ s/(ss|x|sh)es$/$1/ or
      $w =~
s/(virus|fetus|foetus|sinus|lens|atlas|iris|census|genus|penis|focus|callus)es$/$1/ or
      $w =~ s/(hypothes|cris|neuros|psychos|test|synthes|pelv)es$/$1is/ or
      # some special -ies
      $w =~ /(species|caries|facies|series|rabies)$/ or
      $w =~ s/(zombie|calorie|hippie|prairie|movie)s$/$1/ or
      #default -ies -> y
      $w =~ s/ies$/y/ or
      # -ches -> -ch, but NOT niches, avalanches, toothaches, creches, bouches,
douches, psyches, caches etc
      $w =~ s/((ee|[eo]a|ri|r|t|([^a][^l][^a]n)|[^db]ou)ch)es$/$1/ or
      #some special -s to keep
      $w =~
/(propos|always|afterwards|perhaps|whereas|selves|accumbens|meninges|ambiens|annectens
|abducens|oriens|reuniens|ascendens|pancreas|saccharomyces|texas|diabetes|herpes|feces
|faeces|elegans|deferens|kansas|angeles)$/ or
      # default, chop -s
      $w =~ s/s$//;
  }
    }
    return $w;
}
```