Natural Language Processing
CSCI 4152/6509 — Lecture 6
Elements of Morphology

Instructors: Vlado Keselj
Time and date: 16:05 – 17:25, 21-Sep-2023
Location: Rowe 1011

Previous Lecture

- Regular expressions in Perl
  - Use of special variables
  - Backreferences, shortest match
- Text processing examples
  - tokenization
  - counting letters

Letter Frequencies Modification (3)

```
#!/usr/bin/perl
# Letter frequencies (3)

while (<>) {
  while (/[a-zA-Z]/) {
    my $l = $&; $_ = $';
    $f{lc $l} += 1; $tot ++;
  }
}

for (sort { $f{$b} <=> $f{$a} } keys %f) {
  print sprintf("%6d %.4lf %s\n",
                $f{$_}, $f{$_}/$tot, $_); }
```

Output 3

```
35697 0.1204 e
28897 0.0974 t
23528 0.0793 a
23264 0.0784 o
20200 0.0681 n
19608 0.0661 h
18849 0.0635 i
17760 0.0599 s
15297 0.0516 r
14879 0.0502 d
12163 0.0410 l
 8959 0.0302 u
```

...

Elements of Morphology

- Reading: Section 3.1 in the textbook, "Survey of (Mostly) English Morphology"
- *morphemes* — smallest meaning-bearing units
- *stems* and *affixes*; stems provide the "main" meaning, while affixes act as modifiers
- affixes: prefix, suffix, infix, or circumfix
- cliticization — clitics appear as parts of a word, but syntactically they act as words (e.g., 'm, 're, 's)
- tokenization, stemming (Porter stemmer), lemmatization

Tokenization

- Text processing in which plain text is broken into words or *tokens*
- Tokens include non-word units, such as numbers and punctuation
- Tokenization may normalize words by making them lower-case or similar
- Usually simple, but prone to ambiguities, as most of the other NLP tasks

Stemming

- Mapping words to their *stems*
- Example: *foxes* $\rightarrow$ *fox*
- Use in Information Retrieval and Text Mining to normalize text and reduce high dimensionality
- Typically works by removing some suffixes according to a set of rules
- Best known stemmer: Porter stemmer

Lemmatization

- *Surface word form:* a word as it appears in text (e.g., working, are, indices)
- *Lemma:* a canonical or normalized form of a word, as it appears in a dictionary (e.g., work, be, index)
- *Lemmatization:* word processing method which maps surface word forms into their lemmas

Morphological Processes

- Morphological Process = changing word form, as a part of regular language transformation
- Types of morphological processes
  1. inflection
  2. derivation
  3. compounding

1. Inflection

Examples: dog $\rightarrow$ dogs
work $\rightarrow$ works, working, worked

- small change (word remains in the same category)
- relatively regular
- using suffixes and prefixes

2. Derivation

- Typically transforms word in one lexical class to a related word in another class
- Example: wide (adjective) $\rightarrow$ widely (adverb)
  but, similarly: old $\rightarrow$ oldly (*) is incorrect.

  more ex.: accept (verb) $\rightarrow$ acceptable (adjective)
  acceptable (adjective) $\rightarrow$ acceptably (adverb)
  teach (verb) $\rightarrow$ teacher (noun)

- Derivation is a more radical change (change word class)
- less systematic
- using suffixes

Some Derivation Examples

Derivation type	Suffix	Example		
noun-to-verb	*-fy*	glory	$\rightarrow$	glorify
noun-to-adjective	*-al*	tide	$\rightarrow$	tidal
verb-to-noun (agent)	*-er*	teach	$\rightarrow$	teacher
verb-to-noun (abstract)	*-ance*	delivery	$\rightarrow$	deliverance
verb-to-adjective	*-able*	accept	$\rightarrow$	acceptable
adjective-to-noun	*-ness*	slow	$\rightarrow$	slowness
adjective-to-verb	*-ise*	modern	$\rightarrow$	modernise (Brit.)
adjective-to-verb	*-ize*	modern	$\rightarrow$	modernize (U.S.)
adjective-to-adjective	*-ish*	red	$\rightarrow$	reddish
adjective-to-adverb	*-ly*	wide	$\rightarrow$	widely

3. Compounding

Examples:

$$\begin{aligned}
\text{news} + \text{group} &= \text{newsgroup}\\
\text{down} + \text{market} &= \text{downmarket}\\
\text{over} + \text{take} &= \text{overtake}\\
\text{play} + \text{ground} &= \text{playground}\\
\text{lady} + \text{bug} &= \text{ladybug}
\end{aligned}$$

Characters, Words, and N-grams

- We looked at code for counting letters, words, and sentences
- We can look again at counting words; e.g., in "Tom Sawyer":
- We can observe: Zipf's law (1929): $r \times f \approx$ const.

Word	Freq (f)	Rank (r)
the	3331	1
and	2971	2
a	1776	3
to	1725	4
of	1440	5
was	1161	6
it	1030	7
I	1016	8
that	959	9
he	924	10
in	906	11
's	834	12
you	780	13
his	772	14
Tom	763	15
't	654	16
⋮	⋮	

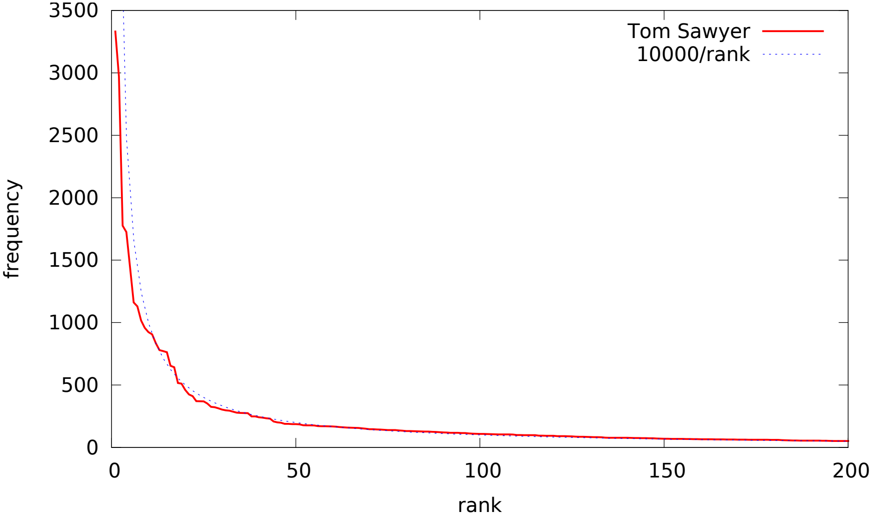
Counting Words

```
#!/usr/bin/perl
# word-frequency.pl

while (<>) {
  while (/'?[a-zA-Z]+/g) { $f{$&}++; $tot++; }
}

print "rank   f   f(norm) word       r*f\n".
      ('-'x35)."\n";
for (sort { $f{$b} <=> $f{$a} } keys %f) {
  print sprintf("%3d. %4d %lf %-8s %5d\n",
             ++$rank, $f{$_}, $f{$_}/$tot, $_,
             $rank*$f{$_});
}
```

Program Output (Zipf's Law)

rank	f	word	r*f
1.	3331	the	3331
2.	2971	and	5942
3.	1776	a	5328
4.	1725	to	6900
5.	1440	of	7200
6.	1161	was	6966
7.	1130	it	7910
8.	1016	I	8128
9.	959	that	8631
10.	924	he	9240
11.	906	in	9966
12.	834	's	10008
13.	780	you	10140
14.	772	his	10808
15.	763	Tom	11445
16.	654	't	10464
17.	642	with	10914
18.	516	for	9288
19.	511	had	9709
20.	460	they	9200
21.	425	him	8925
22.	411	but	9042
23.	371	on	8533
24.	370	The	8880
25.	369	as	9225
26.	352	said	9152
27.	325	He	8775
28.	322	at	9016
29.	313	she	9077
30.	303	up	9090
31.	297	so	9207
32.	294	be	9408
33.	286	all	9438
34.	278	her	9452
35.	276	out	9660
36.	275	not	9900

Graphical Representation of Zipf's Law



Zipf's Law (log-log scale)