CS769 Spring 2010 Advanced Natural Language Processing

Basic Text Process

Lecturer: Xiaojin Zhu *jerryzhu@cs.wisc.edu*

1 Common Preprocessing Steps

Counting words alone gives interesting information. This is known as unigram word count (or word frequency, when normalized). For example Amazon concordance for the book *The Very Hungry Caterpillar* by Eric Carle shows high frequency *content* words "hungry, ate, still, caterpillar, slice, ..." Another interesting website is wordle.net.

How do we count words? First we need to define what a word is. This is highly non-trivial for languages without space like Chinese. Even for seemingly simple English, getting text into a form where you can count words is quite involved, as we see below.

Preprocessing text is called **tokenization** or **text normalization**. Things to consider include

- Throw away unwanted stuff(e.g., HTML tags – but sometimes they are valuable, UUencoding, etc.)
- Word boundaries: white space and punctuations – but words like *Ph.D.*, *isn't*, *e-mail*, *C|net* or *$19.99* are problematic. If you like, we can spend a whole semester on this... This is fairly domain dependent, and people typically use manually created regular expression rules.
- Stemming (Lemmatization): This is optional. English words like 'look' can be *inflected* with a morphological *suffix* to produce 'looks, looking, looked'. They share the same *stem* 'look'. Often (but not always) it is beneficial to map all inflected forms into the stem. This is a complex process, since there can be many exceptional cases (e.g., department vs. depart, be vs. were). The most commonly used *stemmer* is the Porter Stemmer. There are many others. They are not perfect and they do make mistakes. Many are not designed for special domains like biological terms. Some other languages (e.g., Turkish) are particularly hard.
- Stopword removal: the most frequent words often do not carry much meaning. Examples: "the, a, of, for, in, ..." You can also create your own stopword list for your application domain. If we count the words in *Tom Sawyer*, the most frequent word types are (from [MS p.21]):

Word	Count
the	3332
and	2972
a	1775
to	1725
of	1440
was	1161
it	1027
in	906
that	877
he	877

For many NLP purposes (i.e. text categorization) they are a nuisance, and stopword removal is a common preprocessing step. SMART is such a stopword list.

- Capitalization, case folding: often it is convenient to lower case every character. Counterexamples include 'US' vs. 'us'. Use with care.

People devote a large amount of effort to create good text normalization systems.

Now you have clean text, there are two concepts:

- **Word token**: occurrences of a word.
- **Word type**: unique word as a dictionary entry.

For example, "The dog chases the cat." has 5 word tokens but 4 word types. There are two tokens of the word type "the".

A **vocabulary** lists the word types. A typical vocabulary has 10,000 or more words (types). Sometimes also in the vocabulary are `<s>` for the special start-of-sentence symbol, `</s>` for end-of-sentence, and `<unk>` for all out-of-vocabulary words (they all map to this symbol).

A corpus is a large collection of text, e.g., several years' newspapers. A vocabulary can be created from a corpus. Often people apply a *frequency cutoff* to exclude word types with small counts (see below). The cutoff is usually determined empirically (anywhere from one to tens or more).

1.1 Zipf's Law

If we rank word types by their count in *Tom Sawyer*, and compute count $\times$ rank, we see an interesting pattern:

Word	Count f	rank r	fr
the	3332	1	3332
and	2972	2	5944
a	1775	3	5235
he	877	10	8770
but	410	20	8400
be	294	30	8820
there	222	40	8880
one	172	50	8600
two	104	100	10400
turned	51	200	10200
comes	16	500	8000
family	8	1000	8000
brushed	4	2000	8000
Could	2	4000	8000
Applausive	1	8000	8000

We see that $fr \approx$ constant, or $f \propto \frac{1}{r}$. If we plot $\log(r)$ on the x-axis and $\log(f)$ on the y-axis, the words roughly form a line from upper-left to lower-right. Note f can be the frequency (count divided by corpus size) and the relation still holds. This relation is known as *Zipf's law*. It holds for a variety of corpora. Mandelbrot generalizes Zipf's law with more parameters P, ρ, B so it is more flexible: $f = P(r + \rho)^{-B}$.

1.2 Miller's Monkeys

If we promise a monkey some bananas and ask it to type tirelessly on a computer keyboard, what do we get?[1] For simplicity, let us assume the keyboard has 27 keys: a to z, and white space. We also assume the monkey hit each key with equal probability. Let us call a sequence of letters separated by white space a 'word'. What frequency and rank relation do such monkey words possess?

[1] No, not a software engineer.

The probability that a specific monkey word type has length i is

$$P(i) = (1/27)^i(1/27) = (1/27)^{i+1}. \tag{1}$$

As we can see, the longer the word, the lower its probability – therefore the lower the expected count in the monkey corpus. Let us rank all monkey words by its probability. The number of monkey word-types with length i is 26^i. The rank r_i of a word with length i thus satisfies

$$\sum_{j=1}^{i-1} 26^j < r_i \leq \sum_{j=1}^{i} 26^j \tag{2}$$

Let us consider the word with rank $r = \sum_{j=1}^{i} 26^j$. The word actually has length i, but from

$$r = \sum_{j=1}^{i} 26^j = \frac{26}{25}(26^i - 1), \tag{3}$$

we can derive a 'fractional length' i'

$$i' = \frac{\log\left(\frac{25}{26}r + 1\right)}{\log 26}. \tag{4}$$

The frequency of this word is

$$\begin{aligned}
p(i') &= (1/27)^{i'+1} && (5)\\
&= (1/27)^{\frac{\log\left(\frac{25}{26}r+1\right)}{\log 26}+1} && (6)\\
&= (1/27)\left(\frac{25}{26}r + 1\right)^{-\frac{\log 27}{\log 26}} \quad \text{using the fact } a^{\log b} = b^{\log a} && (7)\\
&\approx 0.04(r + 1.04)^{-1.01}, && (8)
\end{aligned}$$

which fits Mandelbrot's law, and is fairly close to Zipf's law.

In light of the above analysis, Zipf's law may not reflect some deep knowledge of languages. Nonetheless, it still points to an important empirical observation, that almost all words are rare. This is also known as the heavy tail property.