Analysis of Language Resources

Sixth Lecture

Hiroyuki Akama

Review

- The quantitative linguistics : the calculation of parameters
- The fundamental index : word frequency, frequency distribution
- Zipf's Laws : r*f=C

$$\frac{F_1}{F_f} = \frac{f(f+1)}{2} \quad If(F_f = 1)thenf = \frac{\sqrt{8F_1 + 1} - 1}{2}$$

- Techniques for gathering frequency data

Programming for Data Gathering

- Shell Script (this time)
- Script Languages as Perl, Ruby, PHP… (especially useful for pattern matching)
- Mathematical power tools as Mathematica, Matlab, MatX… (especially useful for matrix manipulation)
- We will treat this subject next time and in the COE21-LKR practices.

Easy Way to Gather Frequency Data(1)

- **sed**

Stream editor

To erase commas and periods. ←

```
% cat hoge.sed
s/¥.//g
s/¥,//g
% sed –f hoge.sed hoge.txt  >
hoge1.txt
```

- **awk**

program for pattern matching

To provide a dictionary Format (one word in one line). ←

% : Unix command prompt

```
% cat my awk
BEGIN{
FS=” “
}
{
for(i=1;i<=NF;i++)
printf(“%s¥n”, $i)
}
% awk –f myawk hoge1.txt  >hoge2.txt
```

Easy Way to Get Frequency Data(2)

- **sort** (with ***f*** option) : *to sort in alphabetical order*
- **uniq** (with ***c*** option): *to generate a frequency table of multiple lines*→ **% sort –f hoge2.txt | uniq –c >hoge3.txt**

```
#!/bin/csh
#prompt% mycommand hoge.txt
if($#argv[1] !=1) then
echo Usage:header file
exit 1
endif
sed `s/¥.//g ` “$argv[1]”>hoge0.txt
sed `s/¥,//` hoge0.txt >hoge1.txt
awk `BEGIN{FS=” “}{for(i=1;i<=NF;i++) printf(“%s¥n”, $i)}` hoge1.txt >hoge2.txt
sort –f hoge2.txt | uniq –c >hoge3.txt
mv hoge3.txt “$argv[1]”freq.dat
rm hoge?.txt
```

← An example of shell script to get a word frequency table from a plain text ↓

The Features of the Frequency

- Stop Words : Noise Words. The most frequent words (in most cases, *functional* words) that don't characterize documents and thus are often removed for retrieval
- How to treat the stop words:
  - 1) Don't care : Use the *chi-square* values to select content-bearing words
  - 2) Make a general "stop list" including all the standard noise words beforehand
  - 3) Erase the N most frequent words from a particular document

The Frequencies and the Noise (1)

- **The frequency parameters** (from Salton, *The theory of indexing*):
- **The frequency of the term *k* in the document *i*** f_i^k
- **The sum of the frequencies** of the term *k* in the *n* documents (numbered i=1,2,…,n):

$$F^k = \sum_{i=1}^{n} f_i^k$$

- **The importance of the term *k* of the document *i***: $\frac{f_i^k}{F^k}$
- **The document frequency** (the number of the documents with the term *k*)

$$B^k = \sum_{i=1}^{n} b^k$$

$$f_i^k > 0 \text{---} > b_i^k = 1; f_i^k = 0 \text{---} > b_i^k = 0$$

The Frequencies and the Noise (2)

- The signal-noise parameter
- According to the information-theory entropy $H(X) = -\sum_i P(x_i) \log P(x_i) \quad 0 \le H(x) \le \log n$

$$\frac{f_i^k}{F^k} = P(i,k), N^k = -\sum P(i,k) \log P(i,k)$$

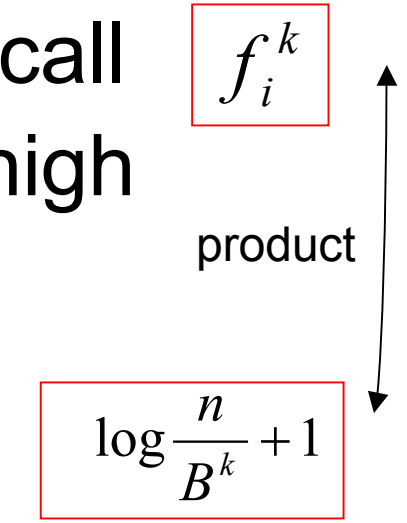
Noise: $N^k = \sum_{i=1}^{n} \frac{f_i^k}{F^k} \log \frac{F^k}{f_i^k}$ Signal: $S^k = \log F^k - N^k$

If the word k appears 1000 times in every document…

White noise or Perfect signal?

Tf,Idf

- Term Frequency (tf): for high recall
- Document Frequency (df) : for high precision
- Inverse Document Frequency (idf) :
- Term Frequency * Inverse Document Frequency (tf.idf) : to estimate how the word contributes to the thematic construction of the document



Parameter Based on Variance

- Variables: $f_i^k, \quad \bar{f}^k = \frac{F^k}{n}$ where $F^k = \sum_{i=1}^{n} f_i^k$
- Variance:

$$(V^k)^2 = \frac{\sum_{i=1}^{n} (f_i^k - \bar{f}^k)^2}{n-1}$$

- EK $$\frac{F^k (V^k)^2}{(\bar{f}^k)^2} = \frac{F^k (V^k)^2}{(\frac{F^k}{n})^2} = \frac{n^2 (V^k)^2}{F^k} = \frac{n}{F^k} \sum_{i=1}^{n} (f_i^k)^2 - F^k$$

- This value is in proportion to variance and in inverse proportion to collection frequency.

Summary

- Techniques to gather frequency data
- Start programming (sed, awk, sort, uniq)
- What to do with stop words
- Parameters for term frequency and document frequency (Signal-noise, tf.idf, Variance)