Natural Language Processing
Regular Expression

BMI701 Introduction of Biomedical Informatics
Lab Session 5

Wei-Hung Weng

October 19, 2016

HMS DBMI — MGH LCS

How?

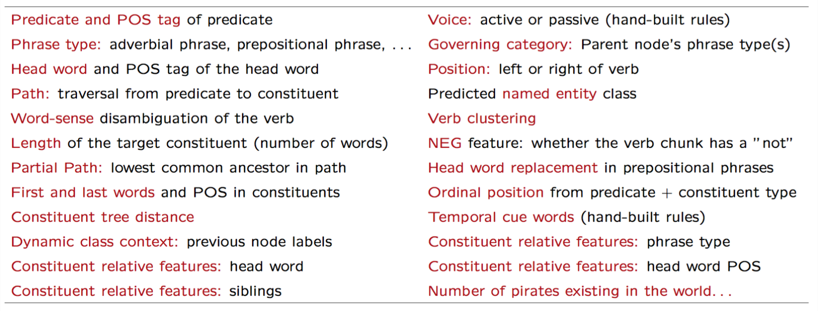
- Collecting / preprocessing data ($>$ 50% of your time)
    - **Regular expression!**
- **NLP with or without linguistic analysis**
- Exploratory analysis, statistics, missing value & outlier
- Annotation and analysis
- Modeling
- Evaluation

Important Feature in Text

- Part-Of-Speech Tagging (POS): syntactic roles (noun, adverb...)
- Chunking (CHUNK): syntactic constituents (noun phrase, verb phrase...)
- Name Entity Recognition (NER): person/company/location...
- Semantic Role Labeling (SRL): semantic role
- Word sense disambiguation (WSD)
- Co-reference resolution

Collobert, Weston 2009

So Many Features



Predicate and POS tag of predicate

Phrase type: adverbial phrase, prepositional phrase, . . .

Head word and POS tag of the head word

Path: traversal from predicate to constituent

Word-sense disambiguation of the verb

Length of the target constituent (number of words)

Partial Path: lowest common ancestor in path

First and last words and POS in constituents

Constituent tree distance

Dynamic class context: previous node labels

Constituent relative features: head word

Constituent relative features: siblings

Voice: active or passive (hand-built rules)

Governing category: Parent node's phrase type(s)

Position: left or right of verb

Predicted named entity class

Verb clustering

NEG feature: whether the verb chunk has a "not"

Head word replacement in prepositional phrases

Ordinal position from predicate + constituent type

Temporal cue words (hand-built rules)

Constituent relative features: phrase type

Constituent relative features: head word POS

Number of pirates existing in the world. . .

Collobert, Weston 2009

Important Feature in Text

- Large scale hand-made feature engineering!
- Task-specific engineering limits NLP scope
- We want to avoid task-specific engineering
- Can we find unified hidden representations? Can we build unified NLP architecture?

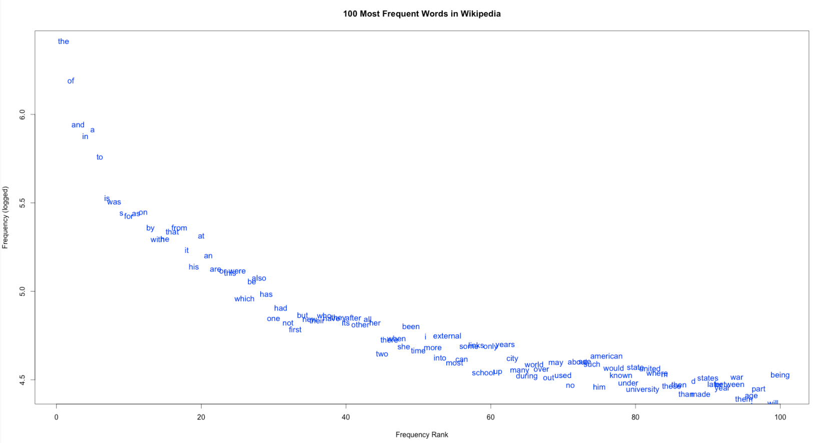
Text Processing

- Text segmentation
  - Alphabetic or Non-alphabetic (Chinese / Japanese / Tibetan...)
  - Separated characters may be meaningless
  - New York-New Haven (the same characters in different order)
- Stemming and Lemmatization (grammar)
  - Different words, same or similar meanings
  - ’imaging’, ’imagination’, ’image’
  - ’be’, ’am’, ’is’, ’are’
- Part-of-speech (POS) tagging
  - NN, VV, ...
  - For semantic analysis
- Stopwords: meaningless
  - Frequent but meaningless or not important

Word Representation Models

- Bag-of-words
  - One-hot encoding representation
  - Simple but useful
  - Frequency $\propto$ representative?
  - Zipf's Law (Zipf 1949)
  - Words with high term frequencies may be just common terms
  - Tf-idf: importance estimation
  - Problem: no word sequence meaning

Zipf's Law



http://wugology.com/zipfs-law/

Word Representation Models

- n-gram model
  - Continuous words
  - Some words are meaningful only when they are observed together
  - Information of word phrase
  - Bag-of-words (n-grams)
    - I like dog
    - BoW: ['I', 'like', 'dog']
    - BoW + n-gram: ['I', 'like', 'dog', 'I like', 'like dog', 'I like dog'] (unigram + bigram + trigram)
- More semantic approach
  - Vectorizing the words
  - Neural word embedding
  - Using neural network to derive vector
  - Compute embedding vectors in a hidden space for words
  - Word2vec (Mikolov 2013)

Text Processing Using R

- `tm` package in R (Feinerer, Hornik 2014)
- Steps
    1. Convert to lower case
    2. Remove punctuation, numbers, URLs, emoji
    3. Remove stopwords
    4. Lemmatization, stemming
    5. Tokenization
    6. POS tagging (optional, not in `tm`)
    7. Convert to document-term matrix

Text Mining

- Wordcloud (`wordcloud`)
- Frequency plot (`ggplot2`)
- Unsupervised learning
    - k-means clustering (`fpc, cluster`)
    - ...
- Supervised learning
    - Decision tree (`rpart`)
    - Support vector machine (`caret`)
    - ...
- github.com/ckbjimmy/bmi701lab/blob/master/lab05.R

Regular Expression

- Crazy regex
- Some tools that can help you
  - regex101
  - regexr
- Regex cheatsheet

Regular Expression

Pattern	Meaning	Example
.	all characters	echocardiogram
cardi	phrase 'cardi'	cardi
.*cardi	0 or more characters before	echocardi
[a-z]*cardi	0+ lower case (only) before	echocardi
[A-Z]*cardi	0+ upper case (only) before	cardi
[aeiou]*cardi	0+ aeiou (only) before	ocardi
[aA-zZ]+cardi	if we use 'xcardiogram'	xcardi
[aA-zZ]{2,}cardi	if we use 'xcardiogram'	-
cardi|gram	catches 'cardi' or 'gram'	cardi, gram
\d	catches any digit	-
\d3, 5	catches 3 to 5 digits	-

- github.com/ckbjimmy/bmi701lab/blob/master/lab05.R

Next Week

- More word representation models
- MetaMap
- cTAKES

Take Home Message

- Bag-of-words
- Regular expression
- Contact
    - Github repository
    - ckbjimmy@gmail.com
    - Linkedin: Wei-Hung Weng