Natural Language

- What is natural language? : Medium of communication
- Forms : text, speech
- Wide spectrum of text : books, newspaper articles, blogs, microblogs and social media posts

Natural language processing (NLP)

- Rules based
- Data driven
- Hybrid (Rules learnt in a data driven manner)

NLP for social media data (SMD)

- With the advent of social media there are large data sets of natural language available for us to study
- Big unstructured natural language data :
  - Volume : 1.3 B monthly active users on FB
  - Velocity : 5700 tweets/sec, 2500 FB posts/sec
  - Variety : Languages, scripts, styles, topics
- Impossible to process manually
- Different kinds of content : tweets / FB posts, reviews, comments, meta data
- Opportunities : SMD is speech-like, personal conversations, language dynamics - evolution of new hashtags, words, spelling changes
- Challenges - Loose grammar, spelling errors, informal evolving vocabulary
- Different kinds of application scenarios : product marketing, political opinion tracking, buzz / trends analysis, sentiment analysis, question answering, summarization, information retrieval and extraction, rumour detection
Basics of processing of SMD

- Tokenization
- Stop Word Removal
- Stemming
- Part of Speech (POS) Tagging
- Named Entity Resolution
- TF-IDF
- Language Detection

References:

- Lecture Slides from the Stanford Coursera course by Dan Jurafsky and Christopher Manning
- Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze, Introduction to Information Retrieval
Tokenization

Given some text, tokenization is the task of chopping it up into pieces called tokens and throwing away certain characters like punctuation.

We have a class on Thursday.

Looks trivial - chop on whitespaces and throw away punctuation. But many tricky cases:

Finland’s capital -> Finland, Finlandsm, Finland’s
What’re, I’m isn’t -> What are, I am, is not
Hewlett-Packard -> Hewlett Packard (?)
State-of-the-art -> state of the art
Lowercase -> lower-case, lowercase, lower case
San francisco -> one or two tokens
Ph.D. -> ?
Email addresses, URLs, @mentions, #tags -> Twitter specific tokenizers
Stop word removal

Some extremely common words that are of little value in helping select documents matching a user's need are excluded from vocabulary entirely

A, an, and, are, as,..., the, that, to, in,...

+ve
Little semantic weight, unlikely to help with retrieval, removing them saves space in inverted index files, terms to doc

-ve
Makes it difficult to search for phrases that contain stop words

To be or not to be -> not
Stemming

Reduce terms to their stems
Crude chopping of affixes or end of words
E.g., automate, automates, automatic, automation -> automat
Porter's Algo for stemming (1980)

Word reductions, applied sequentially conventions to select rules, such as selecting the rule from each rule group that applies to longest suffix

Rule

SSES -> SS  caresses -> caress
IES -> I    ponies -> poni
SS-> SS     caress -> caress
S ->        cats -> cat
(m>1) EMENT -> replacement -> replac cement -> cement

More useful for shorter text, in longer document the different forms are likely to occur

But sometimes throws away useful distinction. E.g., stocks, stockings -> stock
Part of Speech (POS) Tagging

Started with Aristotle in the West (384 - 322 BC), Part of speech = lexical categories, word classes, tags, POS

Thrax of Alexandria (100 BC) - grammatical sketch of greek, 8 parts of speech - noun, verb, articles, adverb, preposition, conjunction, participle, pronoun

English word classes
Closed : relatively fixed membership
Open : nouns and verbs continuously coined or borrowed from other languages

Applications
- Helps in stemming
- Enhance IR app, by selecting alternate nouns or other important words from doc
- Word sense disambiguation
- Named entity recognition
- Sentiment analysis
Input to a tagging algo is a string of words (usually tokenized) and a specific tagset.

Main sources of info for POS tagging:
- Knowledge of neighbouring words
- Knowledge of word probabilities (most frequent tag to word)

Three types computational methods for POS tagging:
- Rule based tagging
- Probabilistic / stochastic methods (Hidden markov model - HMM)
- Transformation based tagging

Rule based tagger

Involves 2 stages
- Uses a dictionary to assign each word a list of potential POS tags
- USEs large lists of hand written disambiguation rules to winnow down this list to single POS for each word
- E.g., an ambiguous word is a noun rather than a verb if it follows a determiner

Hidden Markov Model (HMM)

Generally resolve tagging ambiguities by using a training corpus to compute probability of a given word having a given tag in a given context

Bayesian interpretation
Out of all sequences of n tags \(t^n\), determine the single tag sequence, which is most probable given the observation sequence of n words \(w^n\), i.e., \(P(t^n | w^n)\) is highest

Sequence labeling problem

\[ W^n \rightarrow \text{POS tagger} \rightarrow t^n \]

Our estimate of correct tag sequence, \(t^n\)
\[ t^n = \text{argmax}(t^n) \cdot P(t^n | w^n) \]
Sequence of tags \(t^n\), such that \(P(t^n | w^n)\) is maximized

Using Bayes rule
\[ t^n = \text{argmax}(t^n) \cdot \{ P(w^n | t^n) \cdot P(t^n) \} / \{ P(w^n) \} \]
Drop the denominator because its the same for every \(t^n\) sequence

\[ t^n = \text{argmax}(t^n) \cdot P(w^n | t^n) \cdot P(t^n) \]

\(P(w^n | t^n)\) - Likelihood of word string
\(P(t^n)\) - prior of tag sequence

Simplifying assumptions:

Probability of word appearing is dependent only on its own POS tag.
\[ P(w^n | t^n) = \prod_{i=1}^{n} P(w_i | t_i) = \prod_{i=1}^{n} \frac{\text{count}(t_i, w_i)}{\text{count}(t_i)} \]

Bigram assumption : probability of a tag appearing is dependent only on previous tag.
\[ P(t^n) = \prod_{i=1}^{n} P(t_i | t_{i-1}) = \prod_{i=1}^{n} \frac{\text{count}(t_{i-1}, t_i)}{\text{count}(t_{i-1})} \]

Here count is the number of occurrences in the corpus.
Defined HMM tagging as a task of choosing a tag sequence with the maximum probability derived the equations by which we compute the prob and shown how to compute the component probs.
Decoding algo by which these probs are combined to choose the most likely tag sequence - Viterbi Algo.

Transformation based tagger (Brill tagger)

Like rule based, it is based on rules which determine when an ambiguous word should have a given tag
Like stochastic tagger, it has a machine learning component : rules are automatically induced from a previously tagged training corpus.

Twitter specific CMU POS tagger : [http://www.cs.cmu.edu/~ark/TweetNLP/](http://www.cs.cmu.edu/~ark/TweetNLP/)

Twitter orthography features
Regular expressions style rules to detect @mentions, hashtags, URLs
Frequently capitalized tokens

Traditional tag dictionary : word -> tag

Distributional similarity : used 1.9 M tokens for 134 K unlabelled tweets to construct distributional features from successor and predecessor probs for 10 K most common terms

Phonetic normalization using metaphones
<table>
<thead>
<tr>
<th>Tag</th>
<th>Description</th>
<th>Examples</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>common noun (NN, NNS)</td>
<td>books someone</td>
<td>13.7</td>
</tr>
<tr>
<td>O</td>
<td>pronoun (personal/WH; not possessive; PRP, WP)</td>
<td>you u meeee</td>
<td>6.8</td>
</tr>
<tr>
<td>S</td>
<td>nominal + possessive</td>
<td>books' someone's</td>
<td>0.1</td>
</tr>
<tr>
<td>Z</td>
<td>proper noun + possessive</td>
<td>America's</td>
<td>0.2</td>
</tr>
<tr>
<td>L</td>
<td>nominal + verbal</td>
<td>he's book'll lono</td>
<td>1.6</td>
</tr>
<tr>
<td>M</td>
<td>proper noun + verbal</td>
<td>Mark'll</td>
<td>0.0</td>
</tr>
<tr>
<td>V</td>
<td>verb incl. copula, auxiliaries (Vv, MD)</td>
<td>might gonna</td>
<td>15.1</td>
</tr>
<tr>
<td>A</td>
<td>adjective (J+)</td>
<td>good fav lil</td>
<td>5.1</td>
</tr>
<tr>
<td>R</td>
<td>adverb (R*, XRB)</td>
<td>2 (i.e., too)</td>
<td>4.6</td>
</tr>
<tr>
<td>I</td>
<td>interjection (UH)</td>
<td>lol hafa FTW yea right</td>
<td>2.6</td>
</tr>
<tr>
<td>D</td>
<td>determiner (WDT, DT, WP$, PRP$)</td>
<td>the teh its it's</td>
<td>6.5</td>
</tr>
<tr>
<td>P</td>
<td>pre- or postposition, or</td>
<td>while to for 2 (i.e., so) 4 (i.e., for)</td>
<td>8.7</td>
</tr>
<tr>
<td>&amp;</td>
<td>coordinating conjunction (CC)</td>
<td>and n &amp; + BUT</td>
<td>1.7</td>
</tr>
<tr>
<td>T</td>
<td>verb particle (RP)</td>
<td>out off UP UP</td>
<td>0.6</td>
</tr>
<tr>
<td>X</td>
<td>existential there, predeterminers (EX, PDT)</td>
<td>both</td>
<td>0.1</td>
</tr>
<tr>
<td>Y</td>
<td>X + verbal</td>
<td>there's all's</td>
<td>0.0</td>
</tr>
</tbody>
</table>

**Twitter/online-specific**

| #   | hashtag (indicates topic/category for tweet) | #acl | 1.0 |
| @  | at-mention (indicates another user as a recipient of a tweet) | @BarackObama | 4.9 |
| ~  | discourse marker, indications of continuation of a message across multiple tweets | RT and : in retweet construction RT @user : hello | 3.4 |

**Miscellaneous**

| $   | numeral (CD) | 2010 four 9:30 | 1.5 |
| ,   | punctuation (#, $, ' ', (, ) , , , , ` , `) | !!! .... ??? | 11.6 |
| G   | other abbreviations, foreign words, possessive endings, symbols, garbage (FW, POS, SYM, LS) | (I love you) wby | 1.1 |
|     |             | (what about you)'s --> | |

| E   | emoticon | :-) :b (: <3 o_o | 1.0 |
**Named Entity Recognition (NER)**

Named entity: anything that can be referred to with a proper name

Named entity recognition: Detecting and classifying all the names in a text

<table>
<thead>
<tr>
<th>Type</th>
<th>Tag</th>
<th>Sample Categories</th>
</tr>
</thead>
<tbody>
<tr>
<td>People</td>
<td>PER</td>
<td>Individuals, fictional characters, small groups</td>
</tr>
<tr>
<td>Organization</td>
<td>ORG</td>
<td>Companies, agencies, political parties, religious groups, sports teams</td>
</tr>
<tr>
<td>Location</td>
<td>LOC</td>
<td>Physical extents, mountains, lakes, seas</td>
</tr>
<tr>
<td>Geo-Political Entity</td>
<td>GPE</td>
<td>Countries, states, provinces, counties</td>
</tr>
<tr>
<td>Facility</td>
<td>FAC</td>
<td>Bridges, buildings, airports</td>
</tr>
<tr>
<td>Vehicles</td>
<td>VEH</td>
<td>Planes, trains and automobiles</td>
</tr>
</tbody>
</table>

The course in being taught by Dr. Krishna Gummadi at Saarland University in Saarbrucken.

Ways to signal
- Capitalized words in the middle of the sentence
- Preceded by Dr. or followed by Ph.D. -> name of person
- Facts about proper names and their surrounding context

Uses:
- NE can be indexed, linked
- Sentiment can be attributed to companies or products
- IE relations are associations between NEs.
- For question answering, answers are often NEs.

Two types of ambiguity
- Same name can refer to different entities of same type. JFK - American president or his son
- Identical NEs can refer to entities of completely different types. JFK - person, JFK - facility (airport, schools, bridges, streets named after JFK)

NER as sequence labeling

Word by word sequence labeling task
- Assigned tags capture both boundary and type of any detected NEs.
Representative training doc collection -> human annotations -> annotated docs -> feature extraction and IOB encoding -> training data -> train classifier to perform multiway sequence labeling (HMMs, SVMs, CRFs) -> NER System

Features for sequence labeling

Words
- Current word (like a learned dictionary)
- Previous / next word (context)

Other kinds of inferred linguistic classification : part of speech tags

Label context
- Previous and possibly next label

Application specific name lists

Shape features
- Upper / lower case, capitalised forms. Elaborate patterns for expressions that use number (A9), punctuation (Yahoo!), and atypical case alterations (eBay).
- Useful for formal English
- Not so useful for informally edited sources like blogs, discussion forums
**TF - IDF**

Simple IR technique - Search scenarios.
You have a bunch of docs, if you want to retrieve the most relevant for a query Q.

Compositional semantics
Meaning of a doc resides solely in the set of words it contains. Ordering of words does not matter (syntactic info ignored) => Bag of words models

**Term weighting**

Term frequency $TF_{t,d}$
Number of occurrences of term t in doc d.
Doc that mentions a query term more often has more to do with that query and higher score

Are all words equally important?

Doc frequency $DF_t = \text{number of docs that contains } t$

Inverse doc frequency, $IDF_t = \log N / DF_t$
IDF of rare words is high, IDF of common words is low

Attenuating the effect of terms that occur too often in the collection to be meaningful for relevance determination

$$TF-IDF_{t,d} = TF_{t,d} \times IDF_t$$

Highest when t occurs many times in a small number of docs
Lower when term occurs fewer times in a doc, or occurs in many docs
Lowest when term occurs virtually in all docs (~ stop word removal)
Language Identification

Many different languages on the web

Might want to select posts in one language

Doc level language identification approaches

- Unicode block: different languages use different scripts

- Dictionary based: compute intersection with each language lexicon. Declare highest matching lexicon as winner. Issues: resources intensive, coverage low for short text

- N-gram based techniques
  - Feature: Character n gram (n = 2 to 5)
  - Task: input word(s), output: Yes (belongs to $L_i$)
  - Classifier: Naive Bayes. Max entropy, SVMs
  - Data: Positive egs.: words of $L_i$; Negative egs.: Words from other languages
  - Output: Probability of w being $L_i$.
  - Advantages: Easy to build, robust, easy to bootstrap
  - Issues: Very short noisy text

- Other features
  - Meta data of a webpage
  - User info (social media profile)

Tools: Polyglot, langid.py