Traitements de Corpus — 6

Background Reading:

C. Manning & H. Schütze *Foundations of Statistical Natural Language Processing*, MIT Press, 1999. Chapter 6.

Data sparseness

Training and testing

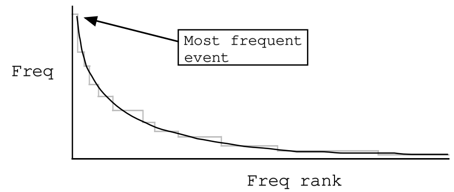
Learning lexical information

Data Sparseness

For n-gram word model (to predict next word), probability function is $P(w_n|w_{n-1}, w_{n-2}, ...)$. If vocabulary 20000 words, numbers of parameters:

bigram: 400 million
trigram: 8 trillion
four-gram: 1.6×10^{17}

Even with smaller numbers of parameters training data may not account for all possible observations (uniform distribution rare). Zipf's Law: inverse exponential relationship between frequency and frequency rank for linguistic categories: $f \propto 1/r$



Must not assign probability zero to events not seen in training data. Techniques:

- forming *equivalence classes* over data instances — stemming, semantic classes (e.g. using WordNet, automatic clustering)
- *smoothing* to re-distribute some probability from seen events to unseen
- *combining estimators*, backing-off to less specific probability models

Equivalence classes

Stemming: non-dictionary-based removal of inflectional and derivational suffixes. Widely used in information retrieval (IR). Document represented by a vector of terms. Terms with same stem usually have similar meanings:

connect, connected, connecting, connection, connections, connectedness

Stemming generally improves IR accuracy. Also reduces total number of terms in system $\longrightarrow$ smaller and more efficient

Smoothing

Maximum likelihood estimation (MLE): $\frac{C}{N}$ $\quad (N =$ total number of events$)$

Slightly decrease probability of seen events so there is some probability mass left over for unseen events

Adding one: $\frac{C+1}{N+B}$ $\quad (B =$ number of distinct events$)$

Expected likelihood estimation (ELE): $\frac{C+\frac{1}{2}}{N+\frac{B}{2}}$

w	$C(w)$	w_1w_2	$C(w_1w_2)$
person	233	person she	2
she	6917	she was	843
was	9409	was inferior	0
inferior	33	inferior to	7
to	20042	to both	9
both	317	both sisters	2

Held out estimation: $\frac{T_C}{N_C N}$ (T_C = number of times all events occurring C times in training data appeared in held out data, and N_C = number of events occurring C times)

Good-Turing smoothing: $f_C^* = (C+1)\frac{N_{C+1}}{N_C}$ (f_C^* = new frequency estimate for event which occurred C times)

Back-off

Deleted interpolation: e.g. mix trigram model with (less sparse) bigram and unigram models

$$\lambda_1 P_1(w_n) + \lambda_2 P_2(w_n|w_{n-1}) + \lambda_3 P_3(w_n|w_{n-1}, w_{n-2})$$

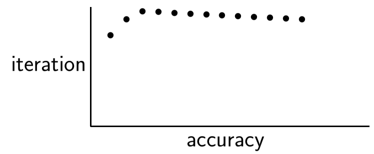
with $0 \leq \lambda_1, \lambda_2, \lambda_3 \leq 1$ and $\lambda_1 + \lambda_2 + \lambda_3 = 1$

Training and Testing

Training on annotated / unannotated data: unannotated (+ dictionary) works OK for tagging, but not well for other higher-level tasks

Overtraining occurs when models are learnt that are tuned to incidental properties of training data set that will lead to errors on any other (unseen) data. E.g.

- decision trees: do not allow to grow to maximum size — can pick version of tree performing best on *held-out data*
- HMMs: Forward-Backward algorithm prone to overtraining especially when applied to annotated data



Primary training data, held-out data, development test set, final test set

Choosing training / test data split

- random choice of small segments
- large chunk of contiguous data
- divide data into n chunks, and train and test on each $n-1$ / 1 split (avoids risk of testing on 'easy' data)

and when comparing different systems use *t test* (if variance is high a difference in accuracy may be purely accidental)

In held-out estimation a relatively large proportion of available data for training is underused since its is used only for smoothing. Use *cross-validation*

- in *deleted estimation*, first make n splits, then combine counts with respect to each split: $\frac{T_C^{1,2}+T_C^{2,1}}{(N_C^1+N_C^2)N}$

Expect much worse accuracy from all statistical techniques if running on a different genre / text type

Learning Lexical Information

Computational lexicons often inadequate:

- missing entries
- missing types of information
- lack of frequency information
- usually general-purpose, not tuned to a particular domain

Lexical acquisition from machine-readable dictionaries (in 1980s)

From corpora (starting 1990s)

- (extended) part-of-speech — *guesser* in tagging systems
- valency — subcategorisation and selectional preferences
- terminology
- bilingual lexicons

Part-of-speech

In Brill's transformation rule-based tagger, initial annotation uses each word's most frequent PoS. What about words that aren't in the lexicon?

Learn separate transformation sequence for unknown words. Applied only to unknown words, before PoS transformations. Initially assign proper noun (NNP) or common noun (NN), depending on capitalization. Rule templates:

For an unknown word *w*, assign tag *A* / change the tag from *A* to *B* if:

- *w* contains character *c*
- *w* has 1/2/3/4 letter prefix *p* (suffix *s*)
- removing prefix *p* (suffix *s*) produces a word that is in the lexicon
- adding prefix *p* (suffix *s*) produces a word that is in the lexicon
- *w* appears to the left (right) of word *x* somewhere in the corpus

Predictive features: starts sentence, contains hyphen / digits?

Subcategorisation

From partially analysed corpora. Shallow analysis → classification → filtering. E.g.

attribute:
... this can not be attributed to the form of publication, only to the limitations of the author.

... this_DD1 can_VM not_XX be_VB0 attribute+ed_VVN to_II the_AT form_NN1 of_IO publication_NN1 ,_, only_RR to_II the_AT limitation+s_NN2 of_IO the_AT author_NN1

obj *iobj (to_II)*

this_DD1 can_VM not_XX be_VB0 attribute+ed_VVN to_II the_AT form_NN1 of_IO publication_NN1 ...

= transitive plus prepositional phrase with nominal complement

Filtering — confidence test on whether a high enough proportion of patterns for class i have been observed for the verb to be in i

Estimate (error) probability $p(v\ \text{-}i)$ of occurrence of verb v not of class i being classified as i (e.g. using frequency of class i in a dictionary, and likelihood of classifier outputting class i)

Probability of an event with probability p happening exactly m times out of n attempts:

$$P(m,n,p) = \frac{n!}{m!(n-m)!}\, p^m (1-p)^{n-m}$$

$$P(m{+},n,p) = \sum_{j=m}^{n} P(j,n,p)$$

$P(m{+},n,p(v\ \text{-}i))$ is probability that m or more occurrences of patterns for class i will occur with a verb which is not a member of i, given n occurrences of that verb

Threshold $\leq$ 0.05 → 95% confidence that 'enough' patterns for i have been observed for the verb to be in class i

Run on 1.2M words from British English corpora; evaluation against large computational lexicons, manual analysis of corpus data

	Merged Entry			Corpus Data			No. of
	TP	FP	FN	TP	FP	FN	Sentences
ask	9	0	18	9	0	10	390
begin	4	1	7	4	1	7	311
believe	4	4	11	4	4	8	230
cause	2	3	6	2	3	5	95
give	5	2	11	5	2	5	639
seem	8	1	4	8	1	4	534
swing	4	0	10	4	0	8	45

Frequency distribution of classes in corpus highly skewed, e.g. for *believe*

- 107 instances of the most common class
- only 6 instances in total of the *least common four* classes

Token recall mean 81% over true positives in the corpus

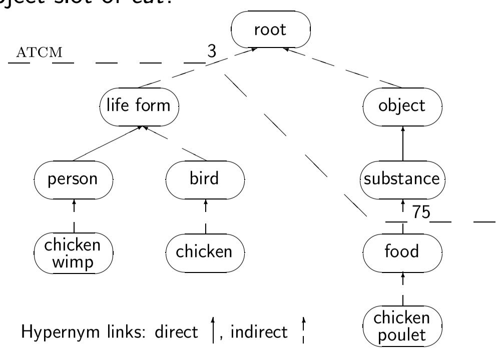
Ranking accuracy of classes mean 81%

Selectional information

Association Tree Cut Model (ATCM)

- preferences represented as set of classes ('tree cut') dominating all leaves in WordNet noun hypernym hierarchy
- 'association score' for each class in cut

E.g. direct object slot of *eat*:



For verbs extract

- subject heads
- noun and preposition-noun argument heads

Minimum Description Length (MDL) principle: select the tree cut model which minimises the sum of

1. the length in bits of the description of the model
2. the length in bits of data encoded in the model