CMPT-882: Statistical Learning of Natural Language

Lecture #13

Anoop Sarkar
anoop@cs.sfu.ca
http://www.sfu.ca/˜anoop

- *Structural Ambiguity and Lexical Relations*. Donald Hindle and Mats Rooth. CL 1993.

- *Statistical models for unsupervised prepositional phrase attachment*. Adwait Ratnaparkhi. COLING-ACL 1998.

- An Unsupervised Approach to Prepositional Phrase Attachment using Contextually Similar Words. P. Pantel and D. Lin. ACL 2000.

Hindle and Rooth 1993

- Basic idea: parse using a minimal commitment parser: where there is ambiguity do not commit to multiple analyses

- Resolve ambiguity using lexical information and pick only one of the multiple analyses

- Data is noisy due to imperfect parsing

- Key difference from (Collins and Brooks 1995) (from last time) is that here the model uses information from parser but is not trained on any data labelled with attachment information

```
The radical changes in export and customs regulations evidently
are aimed at remedying an extreme shortage of consumer goods in
the Soviet Union and assuaging citizens angry over the scarcity
      of such basic items as soap and windshield wipers
```

	Verb	Noun	Prep	Attach
a		changes	in	N
b		regulations		
c	aimed	trace	at	
d	remedying	shortage	of	
e		goods	in	
f		the Soviet Union		
g	assuaging	citizens		
h		scarcity	of	
i		items	as	
j		wipers		

Collecting bigram counts: P,N and P,V

- No preposition: assign NULL – `Items b, f, g, j`
- Sure Verb Attach 1: verb attach preposition if noun is pronoun
- Sure Verb Attach 2: verb attach prep if verb is passive (except if prep is *by*) – `Item c`
- Sure Noun Attach: noun attach prep if verb attach is impossible (e.g. subject noun phrase) – `Item a`

Collecting bigram counts: P,N and P,V

- Ambiguous Attach 1: use likelihood score – `Item d`

- Ambiguous Attach 2: if AA1 does not assign an attachment, split counts between verb and noun attach – `Item d`

- Unsure Attach: assign noun attach – `Items e, h, i` log likelihood ratio: LA >0 is verb attach; <0 is noun attach

- *Moscow sent more than 100,000 soldiers into Afghanistan* ...
  verb attach: [$_{VP}$ send [$_{NP}$ soldier NULL] [$_{PP}$ into ...] ...]
  noun attach: [$_{VP}$ send [$_{NP}$ soldier [$_{PP}$ into ...]] ...]

$$LA(v,n,p) = log_2 \frac{p(a=v,p \mid v,n)}{p(a=n,p \mid v,n)}$$

$$p(a=v, into \mid send, soldier) \approx p(into \mid send) \times p(\text{NULL} \mid soldier)$$

$$p(a=n, into \mid send, soldier) \approx p(into \mid soldier)$$

$$p(into \mid send) = \frac{f(into, send)}{f(send)}$$

$$f(send) = \sum_p f(p, send)$$

Smoothing

$$f(N,p) = \sum_n f(n,p); f(V,p) = \sum_v f(v,p)$$

$$f(N) = \sum_n f(n); f(V) = \sum_v f(v)$$

$$P(p \mid n) = \frac{f(n,p) + \frac{f(N,p)}{f(N)}}{f(n) + 1}$$

$$P(p \mid v) = \frac{f(v,p) + \frac{f(V,p)}{f(V)}}{f(v) + 1}$$

Evaluation

- Right Association (always noun attach): error rate 33%
- Minimal Attachment (always verb attach): error rate 67%
- LA score accuracy: precision 0.797% recall 0.797%
- Comparison with a hand-built dictionary COBUILD: training from text using the LA score outperformed the dictionary (partly because training data for COBUILD was not the same as the test material and partly because of lack of coverage)

Problematic Cases

1. But over time, misery has *given way to* mending `(idiom)`

2. The meeting with *take place in* Quantico `(idiom)`

3. Bush has said he would not *make cuts in* Social Security `(light verb)`

4. Sides said Francke *kept* a .38-caliber *revolver in* his car's glove compartment `(small clause)`

Ratnaparkhi 1998

- Central idea: rather than use a parser to bootstrap training data, use only the unambiguous cases to provide information for the ambiguous ones

- From raw text, use a pos tagger and a chunker and use an extraction heuristic:
  - $(v, p, n2)$ $p \neq$ `of`; v is the first verb K words to the left of p $v \neq$ `to be`; and no noun between v and p
  - $(n, p, n2)$ $p \neq$ `of`; n is the first noun K words to the left of p; and no verb occurs with K words to the left of p

Models

- Heuristics provides 910K attachments (Out of 970K WSJ sents). Accuracy of heuristic $\approx$ 69%

- $cl(v, n, p, n2) = N$ if $p = \texttt{of}$
  $=$ arg $\max_a p(v, n, p, a)$ otherwise

- $p(v, n, p, a) = p(v) \times p(n) \times p(a \mid v, n) \times p(p \mid a, v, n)$

- $p(a = N \mid v, n) = \frac{p(\text{true}|n)}{Z(v,n)}$ (similarly for $a = V$)

Models

- $Z(v, n) = p(\text{true} \mid n) + p(\text{true} \mid v)$

- $p(p \mid a = N, v, n) \approx p(p \mid \text{true}, n)$ (similarly for $a = V$)

- cl_{bigram} using bigram counts:
  $p(p \mid \text{true}, n) = \frac{c(n,p,\text{true})}{c(n,\text{true})}$ if $c(n, \text{true}) > 0$
  $p(p \mid \text{true}, n) = \frac{1}{|\mathcal{P}|}$ otherwise

- cl_{interp} using interpolation smoothing same as Hindle and Rooth 1993.

Evaluation

- No iteration: just sample from raw text and apply on test data
- English: 81.91% cl_{bigram}; 70.39% baseline
- Spanish: 94.5% cl_{bigram}; 90.1% baseline

Pantel and Lin 2000

- find all ambiguous data points: $c(v, p, n2) += \frac{1}{k}$ and $c(n, p, n2) += \frac{1}{k}$ for k attachment sites

- also find all unambiguous data points (as in Ratnaparkhi 1998)

- vscore$(v, p, n2) = ln(p(p)) + ln(p(v, p, n2)) + ln(p(v \mid p)) + ln(p(p \mid n2))$
  nscore$(n, p, n2) = ln(p(p)) + ln(p(n, p, n2)) + ln(p(n \mid p)) + ln(p(p \mid n2))$

Pantel and Lin 2000

- Collocation database:
  eat $\rightarrow$ cook, drink, consume, feed, taste, like, serve, bake, sleep, pick, fry, freeze, . . .
  salad $\rightarrow$ soup, sandwich, sauce, pasta, dish, vegetable, cheese, dessert, entree, bread, meat, chicken, . . .

- Collocation database gives a list of *similar words* to a particular word under consideration

- A *cohort* of a word in a particular **context** is a list of words that appeared in the position of that word in the context
  e.g. in the context of **eat salad** the cohorts of the word *salad* might be *soup, sandwich, pasta, cheese,* . . .

Pantel and Lin 2000

- use the intersection of similar words and cohorts to create an unsupervised alternative to the use of Wordnet in (Stetina and Nagao 1998) find contextually similar words:
  eat with fork $\rightarrow$ spoon, knife, fingers, $\ldots$
  salad with fork $\rightarrow$ $\emptyset$ item use contextually similar words to deal with sparse data: smooth vscore and nscore with counts from this set

- For vscore: given input $(v, n, p, n2)$;
  CS_v is the set of contextually similar words for v in context v : obj : n and
  $CS_{n2,v}$ is the set of csw for $n2$ in context v : p : noun

- define vscore of a set of csw to be the average score for each element in the set

- $\text{vscore}'(v, p, n2) = \max(\text{vscore}(CS_v), \text{vscore}(CS_{n2,v}))$

- similarly: $\text{nscore}'(n, p, n2) = \max(\text{vscore}(CS_n), \text{vscore}(CS_{n,n2}))$

Attachment Algorithm: Pantel and Lin 2000

- **Input**: $(v, n, p, n2)$
- $\text{avg}_n = \text{nscore}'(n, p, n2)$ and $\text{avg}_v = \text{vscore}'(v, p, n2)$
- $a_n = \text{nscore}(n, p, n2)$ and $a_v = \text{vscore}(v, p, n2)$
- $S(n) = \lambda_1 a_n \times \lambda_2 \text{avg}_n$
  $S(v) = \lambda_1 a_v \times \lambda_2 \text{avg}_v$
- **Output**: n if $S(n) > S(v)$ or if $p =$ of; v otherwise

summary of experiments

method	data	accuracy
baseline		70.39
maxent	supervised	81.6
tbl	supervised	81.9
katz (cb)	supervised	84.5
wordnet (sn)	supervised	88.1
parser (hr)	unsupervised	75.8
unamb (r)	unsupervised	81.91
pantel & lin	unsupervised	84.31