CS4025: Statistical NLP, N-grams

Chenghua Lin

Computing Science

Reading: J&M, ch 6 in 1st ed, ch 4 in 2nd

Modelling sequences of words

- Applications for
  - speech disambiguation
  - text classification
  - text generation
  - machine translation

Speech disambiguation

- Choose between similar-sounding words
  - (I scream)/(Ice cream) is delicious
- Possible interpretations
  - I scream is delicious
  - Ice cream is delicious
- Which is more likely to be what the speaker actually said?
  - Based on models of what he/she is likely to say

Traditional AI approach

- Look for interpretation which is
  - grammatical
  - semantically sensible
  - pragmatically plausible in context
- Build explicit models of rules describing how language works
- Poor performance in real world
  - hard to maintain wide-coverage systems
  - too much knowledge needed for semantics

Statistical Approaches

- No explicit models of grammar, meaning, context, etc.
- Instead, collect statistics such as
  - how often different words appear
  - how often pairs (or triples, etc.) of words appear next to each other
  - Or other more complex statistical patterns
- Use this statistics to perform NLP tasks
  - speech, text prediction, translation, ...

N-gram Statistics

- See how often word(s) occur
  - single words (unigram)
  - word pairs (bigram)
  - word triples (trigram)
- Assign a probability to each interpretation, using the statistics
- Pick highest-probability one

Corpus

- Obtain statistics from a *corpus*
  - representative collection of documents
- British National Corpus (BNC)
  - 100M words in 4K documents
  - newspaper articles, academic papers, school essays, memos, radio phone-ins, etc.
  - http://info.ox.ac.uk/bnc
- Google n-gram web corpus
  - 1,024,908,267,229 words

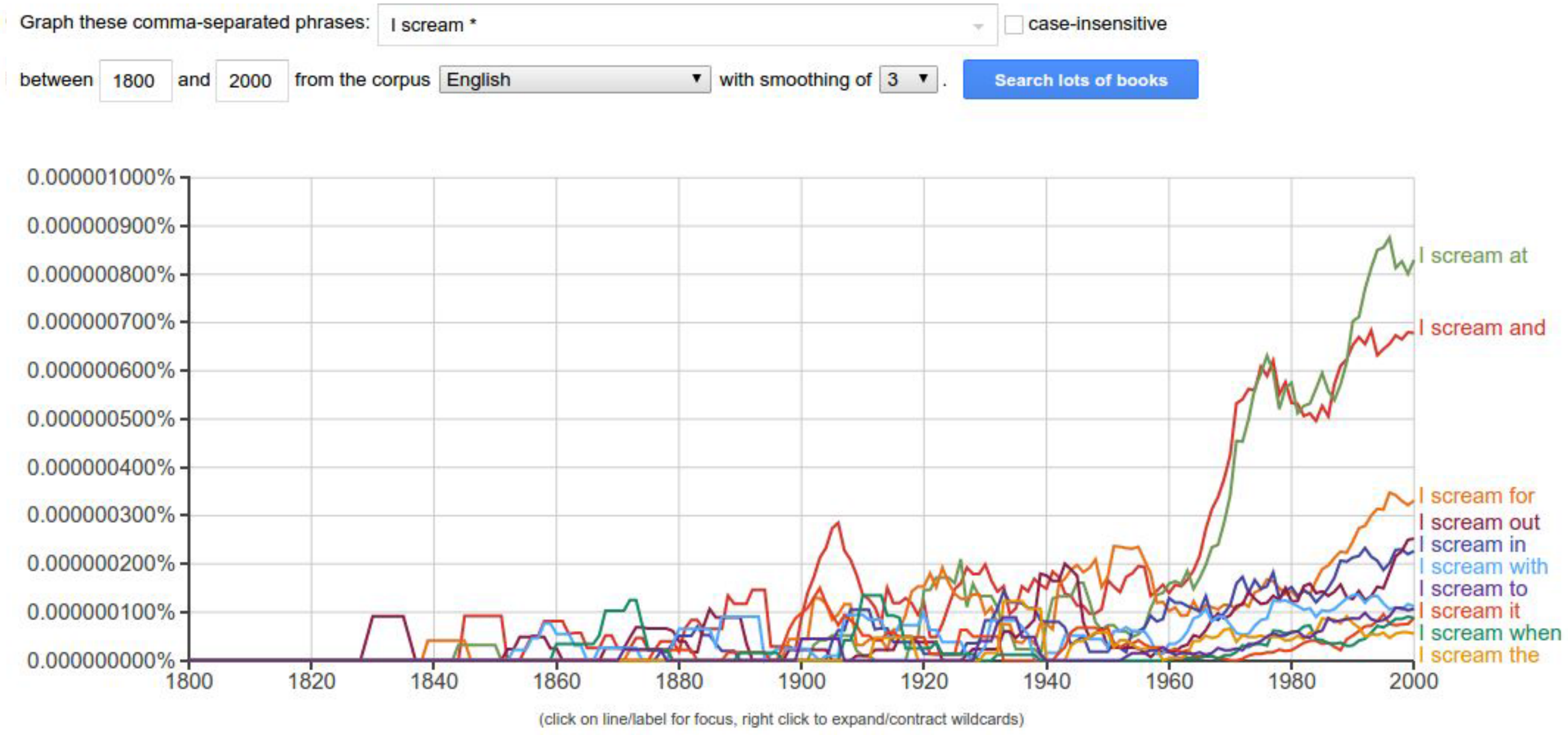
Google Web Corpus

Serve as the (?)

…
serve as the inlet 41
serve as the inner 87
serve as the input 1323
serve as the inputs 189
serve as the insertion 49
serve as the insourced 67
serve as the inspection 43
serve as the inspector 66
serve as the inspiration 1390
serve as the installation 136
serve as the institute 187
serve as the institution 279
serve as the institutional 461
…

Google books n-gram viewer

- Historical n-grams: https://books.google.com/ngrams


Graph these comma-separated phrases: I scream * ☐ case-insensitive

between 1800 and 2000 from the corpus English with smoothing of 3 .

N-gram Language Model

- **Markov assumption**: the probability of a word depends only on the probability of a limited history
- **Generalization**: the probability of a word depends only on the probability of the n previous words

Unigram

- Estimate the probability of each word occurring from its relative frequency in the corpus:
  - $P(w) = \frac{\text{Number of times w occurs}}{\text{Number of words in corpus}}$
- In terms of speech, disambiguation, for each interpretation
  - multiply the estimated unigram probabilities together to get a total probability
- Choose highest probability interpretation

Unigram data

Word	BNC freq	est.prob
<s>	5M	0.2
I	900K	9E-3 (.009)
ice	4K	4E-5 (.00004)
scream	1K	1E-5 (.00001)
cream	3K	3E-5 (.00003)
is	1M	1E-2 (.01)
delicious	1K	1E-5 (.00001)

<s> is “start of sentence”. Frequencies involving <s> are invented.

Total probability

- (1) Ice cream is delicious
  - P(ice) * P(cream) * P(is) * P(delicious)
  - 4E-5 * 3E-5 * 1E-2 * 1E-5 = 1.2E-16 (= 12E-17)
- (2) I scream is delicious
  - P(I) * P(scream) * P(is) * P(delicious)
  - 9E-3 * 1E-5 * 1E-2 * 1E-5 = 9 E-15
- (2) has highest probability
- But this **ignores the order of the words**!

Generating sentences using the Unigram Model

- Random sentences with words selected according to their unigram probabilities (based on a corpus of airline reservation queries):
  - Of aircraft fare east
  - The the stopover city the coach the one_way frequent in travelling flights
  - Two on flight really Boston me like
  - And like six would these fifty

Bigram Language Model

- Bigram probability:

$$P(w_1, w_2) = \frac{\text{number of times "}w_1\ w_2\text{" appears}}{\text{number of times any bigram occurs}}$$

Bigram Probability

- $P(w_2|w_1) = P(w_1, w_2)/P(w_1)$ (Bayes rule)

Unigram Probability

- Assign each interpretation a total probability of sequence based on conditional probabilities:
  - $P(w_1 \dots w_n) = P(w_1|\text{<s>}) * P(w_2|w_1) * \dots P(w_n|w_{n-1})$
- Choose interpretation with highest likelihood

Bigram data

Bigram	BNC freq
<s> I	800K
<s> Ice	400
I scream	9
Ice cream	454
scream is	4
cream is	40
is delicious	36

Frequency of “I” is 900K. So P(scream|I) = 9/900K

Total score

- (1) Ice cream is delicious
  - P(ice|<s>) * P(cream|ice) * P(is|cream) * P(delicious|is)
  - (400/5M) * (454/4K) * (40/3K) * (36/1M)
- (2) I scream is delicious
  - P(I|<s>) * P(scream|I) * P(is|scream) * P(delicious|is)
  - (800K/5M) * (9/900K) * (4/1K) * (36/1M)
- Choose the one which has highest score

Generate sentences using a bigram model

- Selecting random sentences based on the airline corpus using bigram probabilities:
  - I want a first class
  - I would like to San_Francisco
  - What about twelve and Baltimore
  - Are the next week on AA eleven ten
  - In Denver on October
  - What is the city of three pm

Trigrams

- Trigrams: use word triples
- Need a **huge** corpus to get good stats

Trigram	BNC freq
<s1> <s2> I	800K
<s1> <s2> ice	400
<s2> I scream	4
<s2> ice cream	138
I scream is	0
ice cream is	10
scream is delicious	0
cream is delicious	0

Total score

- (1) Ice cream is delicious
  - P(ice|<s1><s2>) * P(cream|<s2> ice) * P(is|ice cream) * P(delicious|cream is)
  - (400/5M) * (138/400) * (10/454) * (0/40)
  - = 0
- (2) I scream is delicious
  - P(I|<s1><s2>) * P(scream|<s2> I) * P(is|I scream) * P(delicious|scream is)
  - (800K/5M) * (4/800K) * (0/9) * (0/4)
  - = 0
- Can’t distinguish

Generating sentences using a trigram model

- Random sentences using trigram probabilities:
  - How much does it cost
  - I'd like to depart before five o'clock pm
  - How many stops does Delta flight five eleven o'clock pm that go from what am
  - I need to Philadelphia
  - Describe to Baltimore on Wednesday from Boston
  - Which flight do these flights leave after four pm and lunch and <unk>

Smoothing

- If an N-gram doesn't occur at all, give it a small weight instead of 0
- Simplest approach (*add one smoothing*): imagine there is one extra document in the corpus which contains each possible N-gram exactly once.
- For bigrams, this imaginary document will contain each possible word V times, where V is the number of different words.

$$P(w_n|w_{n-1}) = \frac{c(w_{n-1,}w_n)}{c(w_{n-1})}$$

$$P(w_n|w_{n-1}) = \frac{c(w_{n-1,}w_n)+1}{c(w_{n-1})+V}$$

Corpus Issues

- Corpus should be similar to intended texts
  - If corpus is Wall Street Journal, don’t expect good performance on social chat
- Get funny results
  - Prednisolone is more common in the BNC than Paracetamol
    - because BNC collection includes a few medical research papers which talk about Prednisolone

Summary

- N-gram models
  - Based on word (singleton, pair, triple, etc) frequencies in a corpus
  - Very effective at many tasks, **if** enough data to estimate frequencies reliably
    - Smoothing techniques can increase reliability where frequencies are very low (not just 0)
    - Also known as the *sparse data* problem
  - No need to explicitly program grammars, build huge AI knowledge bases, etc.
    - models are learned, not programmed