Statistical NLP
Spring 2009

Lecture 3: Language Models II

Dan Klein – UC Berkeley

Puzzle: Unknown Words

- Imagine we look at 1M words of text
  - We'll see many thousands of word types
  - Some will be frequent, others rare
  - Could turn into an empirical P(w)
- Questions:
  - What fraction of the next 1M will be new words?
  - How many total word types exist?

Language Models

- In general, we want to place a distribution over sentences
- Basic / classic solution: n-gram models

$$P(w) = \prod_i P(w_i | w_{i-1} \ldots w_{i-k})$$

- Question: how to estimate conditional probabilities?

$$P(w|w') =$$

- Problems:
  - Known words in unseen contexts
  - Entirely unknown words
    - Many systems ignore this – why?
    - Often just lump all new words into a single UNK type

Smoothing: Add-One, Etc.

- With a uniform prior, get estimates of the form

$$P_{\text{add}-\delta}(x) = \frac{c(x) + \delta}{\sum_{x'}(c(x') + \delta)}$$

  - Add-one smoothing especially often talked about
- For a bigram distribution, can use a prior centered on the empirical unigram:

$$P_{dir}(w|w_{-1}) = \frac{c(w_{-1}, w) + k\hat{P}(w)}{\left(\sum_{w'} c(w_{-1}, w')\right) + k}$$

- Can consider hierarchical formulations: trigram is recursively centered on smoothed bigram estimate, etc [MacKay and Peto, 94]
- Basic idea of conjugacy is convenient: prior shape shows up as *pseudo-counts*
- Problem: works quite poorly!

Linear Interpolation

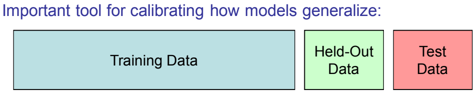
- Problem: $\hat{P}(w|w_{-1}, w_{-2})$ is supported by few counts
- Classic solution: mixtures of related, denser histories, e.g.:

$$\lambda \hat{P}(w|w_{-1}, w_{-2}) + \lambda' \hat{P}(w|w_{-1}) + \lambda'' \hat{P}(w)$$

- The mixture approach tends to work better than the Dirichlet prior approach for several reasons
  - Can flexibly include multiple back-off contexts, not just a chain
  - Often multiple weights, depending on bucketed counts
  - Good ways of learning the mixture weights with EM (later)
  - Not entirely clear why it works so much better
- All the details you could ever want: [Chen and Goodman, 98]

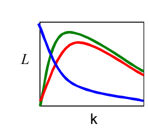
Held-Out Data

- Important tool for calibrating how models generalize:



  - Set a small number of hyperparameters that control the degree of smoothing by maximizing the (log-)likelihood of held-out data
  - Can use any optimization technique (line search or EM usually easiest)
- Examples:

$$P_{dir}(w|w_{-1}, k) = \frac{c(w_{-1}, w) + k\hat{P}(w)}{\left(\sum_{w'} c(w_{-1}, w')\right) + k}$$



$$P_{lin}(w|w_{-1}, \lambda, \lambda', \lambda'') = \lambda \hat{P}(w|w_{-1}, w_{-2}) + \lambda' \hat{P}(w|w_{-1}) + \lambda'' \hat{P}(w)$$

Held-Out Reweighting

- What's wrong with unigram-prior smoothing?
- Let's look at some real bigram counts [Church and Gale 91]:

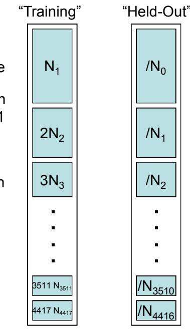
Count in 22M Words	Actual c* (Next 22M)	Add-one's c*	Add-0.0000027's c*
1	0.448	2/7e-10	~1
2	1.25	3/7e-10	~2
3	2.24	4/7e-10	~3
4	3.23	5/7e-10	~4
5	4.21	6/7e-10	~5

Mass on New	9.2%	~100%	9.2%
Ratio of 2/1	2.8	1.5	~2

- Big things to notice:
  - Add-one vastly overestimates the fraction of new bigrams
  - Add-0.0000027 vastly underestimates the ratio 2*/1*
- One solution: use held-out data to predict the map of c to c*

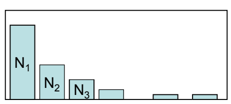
Good-Turing Reweighting I

- We'd like to not need held-out data (why?)
- Idea: leave-one-out validation
  - N_k: number of types which occur k times in the entire corpus
  - Take each of the c tokens out of corpus in turn
  - c "training" sets of size c-1, "held-out" of size 1
  - How many held-out tokens are unseen in training?
    - N_1
  - How many held-out tokens are seen k times in training?
    - $(k+1)N_{k+1}$
  - There are N_k words with training count k
  - Each should occur with expected count
    - $(k+1)N_{k+1}/N_k$
  - Each should occur with probability:
    - $(k+1)N_{k+1}/(cN_k)$

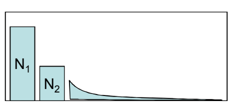


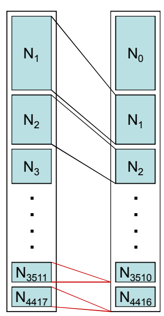
Good-Turing Reweighting II

- Problem: what about "the"? (say k=4417)
  - For small k, $N_k > N_{k+1}$
  - For large k, too jumpy, zeros wreck estimates



  - Simple Good-Turing [Gale and Sampson]: replace empirical N_k with a best-fit power law once count counts get unreliable





Good-Turing Reweighting III

- Hypothesis: counts of k should be $k^* = (k+1)N_{k+1}/N_k$

Count in 22M Words	Actual c* (Next 22M)	GT's c*
1	0.448	0.446
2	1.25	1.26
3	2.24	2.24
4	3.23	3.24
Mass on New	9.2%	9.2%

- Katz Smoothing
  - Use GT discounted *bigram* counts (roughly – Katz left large counts alone)
  - Whatever mass is left goes to empirical unigram

$$P_{\text{katz}}(w|w') = \frac{c^*(w',w)}{c(w')} + \alpha(w')\hat{P}(w)$$

Kneser-Ney: Discounting

- Kneser-Ney smoothing: very successful but slightly ad hoc estimator
- Idea: observed n-grams occur more in training than they will later:

Count in 22M Words	Avg in Next 22M	Good-Turing c*
1	0.448	0.446
2	1.25	1.26
3	2.24	2.24
4	3.23	3.24

- Absolute Discounting
  - Save ourselves some time and just subtract 0.75 (or some d)
  - Maybe have a separate value of d for very low counts

$$P_{\text{ad}}(w|w') = \frac{c(w',w) - d}{c(w')} + \alpha(w')\hat{P}(w)$$

Kneser-Ney: Continuation

- Something's been very broken all this time
  - Shannon game: There was an unexpected ____?
    - delay?
    - Francisco?
  - "Francisco" is more common than "delay"
  - … but "Francisco" always follows "San"
- Solution: Kneser-Ney smoothing
  - In the back-off model, we don't want the probability of w as a unigram
  - Instead, want the probability that w is *allowed in this novel context*
  - For each word, count the number of bigram types it completes

$$P_{\text{C}}(w) \propto |w' : c(w',w) > 0|$$

Kneser-Ney

- Kneser-Ney smoothing combines these two ideas
  - Absolute discounting

$$P(w|w') = \frac{c(w',w) - d}{c(w')} + \alpha(w')P'(w)$$

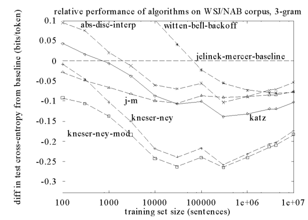
  - Lower order models take a special form

$$P_c(w) \propto |w' : c(w',w) > 0|$$

- KN smoothing repeatedly proven effective
  - But we've never been quite sure why
  - And therefore never known how to make it better
- [Teh, 2006] shows KN smoothing is a kind of approximate inference in a hierarchical Pitman-Yor process (and better approximations are superior to basic KN)

What Actually Works?

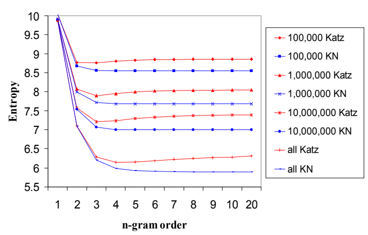
- Trigrams:
  - Unigrams, bigrams too little context
  - Trigrams much better (when there's enough data)
  - 4-, 5-grams often not worth the cost (which is more than it seems, due to how speech recognizers are constructed)
  - Note: for MT, 5+ often used!
- Good-Turing-like methods for count adjustment
  - Absolute discounting, Good-Turing, held-out estimation, Witten-Bell
- Kneser-Ney equalization for lower-order models
- See [Chen+Goodman] reading for tons of graphs!



[Graphs from Joshua Goodman]

Data >> Method?

- Having more data is better…



- … but so is using a better model
- Another issue: N > 3 has huge costs in speech recognizers

Beyond N-Gram LMs

- Lots of ideas we won't have time to discuss:
  - Caching models: recent words more likely to appear again
  - Trigger models: recent words trigger other words
  - Topic models
- A few recent ideas
  - Syntactic models: use tree models to capture long-distance syntactic effects [Chelba and Jelinek, 98]
  - Discriminative models: set n-gram weights to improve final task accuracy rather than fit training set density [Roark, 05, for ASR; Liang et. al., 06, for MT]
  - Structural zeros: some n-grams are syntactically forbidden, keep estimates at zero [Mohri and Roark, 06]
  - Bayesian document and IR models [Daume 06]

Overview

- So far: language models give P(s)
  - Help model fluency for various noisy-channel processes (MT, ASR, etc.)
  - N-gram models don't represent any deep variables involved in language structure or meaning
  - Usually we want to know something about the input other than how likely it is (syntax, semantics, topic, etc)
- Next: Naïve-Bayes models
  - We introduce a single new global variable
  - Still a very simplistic model family
  - Lets us model hidden properties of text, but only very non-local ones…
  - In particular, we can only model properties which are largely invariant to word order (like topic)

Text Categorization

- Want to classify documents into broad semantic topics (e.g. politics, sports, etc.)

Obama is hoping to rally support for his $825 billion stimulus package on the eve of a crucial House vote. Republicans have expressed reservations about the proposal, calling for more tax cuts and less spending. GOP representatives seemed doubtful that any deals would be made.

California will open the 2009 season at home against Maryland Sept. 5 and will play a total of six games in Memorial Stadium in the final football schedule announced by the Pacific-10 Conference Friday. The original schedule called for 12 games over 12 weekends.

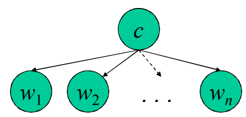
- Which one is the politics document? (And how much deep processing did that decision take?)
- One approach: bag-of-words and Naïve-Bayes models
- Another approach later…
- Usually begin with a labeled corpus containing examples of each class

Naïve-Bayes Models

- Idea: pick a topic, then generate a document using a language model for that topic.
- Naïve-Bayes assumption: all words are independent given the topic.

$$P(c, w_1, w_2, \ldots w_n) = P(c)\prod_i \boxed{P(w_i \mid c)}$$

We have to smooth these!



- Compare to a unigram language model:

$$P(w_1, w_2, \ldots w_n) = \prod_i P(w_i)$$

Using NB for Classification

- We have a joint model of topics and documents

$$P(c, w_1, w_2, \ldots w_n) = P(c)\prod_i P(w_i \mid c)$$

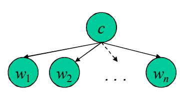
- Gives posterior likelihood of topic given a document

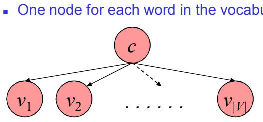
$$P(c \mid w_1, w_2, \ldots w_n) = \frac{P(c)\prod_i P(w_i \mid c)}{\sum_{c'}\left[P(c')\prod_i P(w_i \mid c')\right]}$$

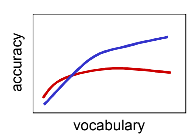
- What about totally unknown words?
- Can work shockingly well for textcat (especially in the wild)
- How can unigram models be so terrible for language modeling, but class-conditional unigram models work for textcat?
- Numerical / speed issues
- How about NB for spam detection?

Two NB Formulations

- Two NB event models for text categorization
  - The class-conditional unigram model, a.k.a. multinomial model
    - One node per word in the document
    - Driven by words which are present
    - Multiple occurrences, multiple evidence
    - Better overall – plus, know how to smooth
  - The binomial (binary) model
    - One node for each word in the vocabulary







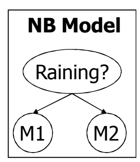
    - Incorporates explicit negative correlations
    - Know how to do feature selection (e.g. keep words with high mutual information with the class variable)

Example: Barometers

Reality

Raining | Sunny

P(+,+,r) = 1/8 | P(-,-,r) = 3/8 | P(+,+,s) = 3/8 | P(-,-,s) = 1/8



NB FACTORS:
- P(s) = 1/2
- P(-|s) = 1/4
- P(-|r) = 3/4

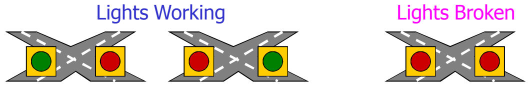
PREDICTIONS:
- P(r,-,-) = (½)(¾)(¾)
- P(s,-,-) = (½)(¼)(¼)
- P(r|-,-) = 9/10
- P(s|-,-) = 1/10

Overconfidence!

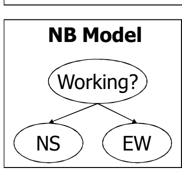
Example: Stoplights

Reality

Lights Working | Lights Broken



P(g,r,w) = 3/7 | P(r,g,w) = 3/7 | P(r,r,b) = 1/7

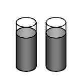


NB FACTORS:
- P(w) = 6/7
- P(r|w) = 1/2
- P(g|w) = 1/2
- P(b) = 1/7
- P(r|b) = 1
- P(g|b) = 0

P(b|r,r) = 4/10 (what happened?)

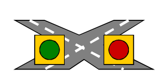
(Non-)Independence Issues

- Mild Non-Independence
  - Evidence all points in the right direction
  - Observations just not entirely independent
  - Results
    - Inflated Confidence
    - Deflated Priors
  - What to do? Boost priors or attenuate evidence



$$P(c, w_1, w_2, \ldots w_n) \text{ "=" } P(c)^{boost>1}\prod_i P(w_i \mid c)^{boost<1}$$

- Severe Non-Independence
  - Words viewed independently are misleading
  - Interactions have to be modeled
  - What to do?
    - Change your model!



Language Identification

- How can we tell what language a document is in?

The 38th Parliament will meet on Monday, October 4, 2004, at 11:00 a.m. The first item of business will be the election of the Speaker of the House of Commons. Her Excellency the Governor General will open the First Session of the 38th Parliament on October 5, 2004, with a Speech from the Throne.

La 38e législature se réunira à 11 heures le lundi 4 octobre 2004, et la première affaire à l'ordre du jour sera l'élection du président de la Chambre des communes. Son Excellence la Gouverneure générale ouvrira la première session de la 38e législature avec un discours du Trône le mardi 5 octobre 2004.

- How to tell the French from the English?
  - Treat it as word-level textcat?
    - Overkill, and requires a lot of training data
    - You don't actually need to know about words!

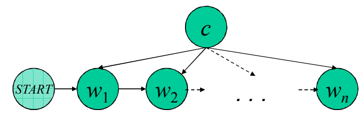
Σύμφωνο σταθερότητας και ανάπτυξης
Patto di stabilità e di crescita

  - Option: build a character-level language model

Class-Conditional LMs

- Can add a topic variable to other language models

$$P(c, w_1, w_2, \ldots w_n) = P(c)\prod_i P(w_i \mid w_{i-1}, c)$$



- Could be characters instead of words, used for language ID (HW2)
- Could sum out the topic variable and use as a language model
- How might a class-conditional n-gram language model behave differently from a standard n-gram model?