CS114 Lecture 4

Some more about words

Probabilities

Ngrams

January 28, 2013

Professor Meteer

Thanks for Jurafsky & Martin & Prof. Pustejovksy for slides

From Counting to Probabilities

- Why count words
- A probability primer
- Ngrams and Language models

Word Prediction

- Guess the next word...
  - ... *I notice three guys standing on the ???*
- There are many sources of knowledge that can be used to inform this task, including arbitrary world knowledge.
- But it turns out that you can do pretty well by simply looking at the preceding words and keeping track of some fairly simple counts.

Statistical Example

The preceding and following words lend valuable information to the probability of the whole.

What is the missing word?

"I love _______"

NOW what is the missing word?

"My favorite TV show was I love _______"

Word Prediction

- We can formalize this task using what are called *N*-gram models.
- *N*-grams are token sequences of length *N*.
- Our earlier example contains the following 2-grams (aka bigrams)
  - (I notice), (notice three), (three guys), (guys standing), (standing on), (on the)
- Given knowledge of counts of N-grams such as these, we can guess likely next words in a sequence.

N-Gram Models

- More formally, we can use knowledge of the counts of *N*-grams to assess the conditional probability of candidate words as the next word in a sequence.
- Or, we can use them to assess the probability of an entire sequence of words.
  - Pretty much the same thing as we'll see...

Applications

- It turns out that being able to predict the next word (or any linguistic unit) in a sequence is an extremely useful thing to be able to do.
- As we'll see, it lies at the core of the following applications
  - Automatic speech recognition
  - Handwriting and character recognition
  - Spelling correction
  - Machine translation
  - And many more.

Counting

- Simple counting lies at the core of any probabilistic approach. So let's first take a look at what we're counting.
  - *He stepped out into the hall, was delighted to encounter a water brother.*
    - 13 tokens, 15 if we include "," and "." as separate tokens.
    - Assuming we include the comma and period, how many bigrams are there?

Counting

- Not always that simple
  - *I do uh main- mainly business data processing*
- Spoken language poses various challenges.
  - Should we count “uh” and other fillers as tokens?
  - What about the repetition of “mainly”? Should such do-overs count twice or just once?
  - The answers depend on the application.
    - If we’re focusing on something like ASR to support indexing for search, then “uh” isn’t helpful (it’s not likely to occur as a query).
    - But filled pauses are very useful in dialog management, so we might want them there.

Counting: Types and Tokens

- How about
  - *They picnicked by the pool, then lay back on the grass and looked at the stars.*
    - 18 tokens (again counting punctuation)
- But we might also note that "*the*" is used 3 times, so there are only 16 unique types (as opposed to tokens).
- In going forward, we'll have occasion to focus on counting both types and tokens of both words and *N*-grams.

Counting: Wordforms

- Should “cats” and “cat” count as the same when we’re counting?
- How about “geese” and “goose”?
- Some terminology:
  - Lemma: a set of lexical forms having the same stem, major part of speech, and rough word sense
  - Wordform: fully inflected surface form
- Again, we’ll have occasion to count both lemmas and wordforms

Counting: Corpora

- So what happens when we look at large bodies of text instead of single utterances?
- Brown et al (1992) large corpus of English text
  - 583 million wordform tokens
  - 293,181 wordform types
- Google
  - Crawl of 1,024,908,267,229 English tokens
  - 13,588,391 wordform types
    - That seems like a lot of types... After all, even large dictionaries of English have only around 500k types. Why so many here?

- Numbers
- Misspellings
- Names
- Acronyms
- etc

Language Modeling

- Back to word prediction
- We can model the word prediction task as the ability to assess the conditional probability of a word given the previous words in the sequence
  - $P(w_n|w_1,w_2\ldots w_{n-1})$
- We'll call a statistical model that can assess this a *Language Model*

Introduction to Probability

- Experiment (trial)
  - Repeatable procedure with well-defined possible outcomes
- Sample Space (S)
    - the set of all possible outcomes
    - *finite or infinite*
  - *Example*
    - *coin toss experiment*
    - *possible outcomes: S = {heads, tails}*
  - *Example*
    - *die toss experiment*
    - *possible outcomes: S = {1,2,3,4,5,6}*

Introduction to Probability

- Definition of sample space depends on what we are asking
  - Sample Space (S): the set of all possible outcomes
  - Example
    - die toss experiment for whether the number is even or odd
    - possible outcomes: {even,odd}
    - *not {1,2,3,4,5,6}*

More Definitions

- Events
  - an ***event*** is ***any subset of outcomes from the sample space***
- Example
  - ***die toss experiment***
  - ***let A represent the event such that the outcome of the die toss experiment is divisible by 3***
  - ***A = {3,6}***
  - ***A is a subset of the sample space S= {1,2,3,4,5,6}***
- Example
  - ***Draw a card from a deck***
    - ***suppose sample space S = {heart,spade,club,diamond} (four suits)***
    - ***let A represent the event of drawing a heart***
    - ***let B represent the event of drawing a red card***
    - ***A = {heart}***
    - ***B = {heart,diamond}***

Definition of Probability

- The probability law assigns to an event a nonnegative number
  - Called P(A) or the probability A
- That encodes our knowledge or belief about the collective likelihood of all the elements of A

Laws of Probability

- Probability law must satisfy certain properties
  - Nonnegativity
    - P(A) >= 0, for every event A
  - Additivity
    - If A and B are two disjoint events, then the probability of their union satisfies: P(A U B) = P(A) + P(B)
  - Normalization
    - The probability of the entire sample space S is equal to 1,
    - i.e. P(S) = 1.

An example

- Experiment involving 3 coin tosses
- Outcome is a 3-long string of H or T S

  ={HHH, HHT, HTH, HTT, THH, THT, TTH, TTT}
- Assume each outcome is equi-probable
  - "Uniform distribution"
- What is probability of the event that exactly 2 heads occur?

  A = {HHT,HTH,THH}

  P(A) = P({HHT})+P({HTH})+P({THH})

  = 1/8 + 1/8 + 1/8

  = 3/8

In Summary

$$P(E) = \frac{\text{Number of outcomes corresponding to event E}}{\text{Total number of outcomes}}$$

- Probability of drawing a spade from 52 well-shuffled playing cards:

$$\frac{13}{52} = \frac{1}{4} = .25$$

Probabilities of two events

- If two events A and B are independent
  - Then
    - P(A and B) = P(A) x P(B)
- If flip a fair coin twice
  - What is the probability that they are both heads?
- If draw a card from a deck, then put it back, draw a card from the deck again
  - What is the probability that both drawn cards are hearts?

.25 x .25 = .0625

How about non-uniform probabilities?

- A biased coin,
  - twice as likely to come up tails as heads,
  - is tossed twice
- What is the probability that at least one head occurs?
- Sample space = {hh, ht, th, tt}
- Sample points/probability for the event:

  ht 1/3 x 2/3 = **2/9**

  hh 1/3 x 1/3= 1/9

  th 2/3 x 1/3 = 2/9

  tt 2/3 x 2/3 = 4/9

- **Answer: 5/9 = ≈0.56 (sum of weights in red)**

Moving toward language

- What's the probability of drawing a 2 from a deck of 52 cards with four 2s?

$$P(\text{drawing a two}) = \frac{4}{52} = \frac{1}{13} = .077$$

- What's the probability of a random word (from a random dictionary page) being a verb?

$$P(\text{drawing a verb}) = \frac{\text{\#of ways to get a verb}}{\text{all words}}$$

Probability and part of speech tags

- What’s the probability of a random word (from a random dictionary page) being a verb?

$$\text{P(picking a verb)} = \frac{\textit{\#of ways to get a verb}}{\textit{all words}}$$

- How to compute each of these
  - All words = just count all the words in the dictionary
  - # of ways to get a verb: number of words which are verbs!
  - If a dictionary has 50,000 entries, and 10,000 are verbs…. P(V) is 10000/50000 = 1/5 = .20

Conditional Probability

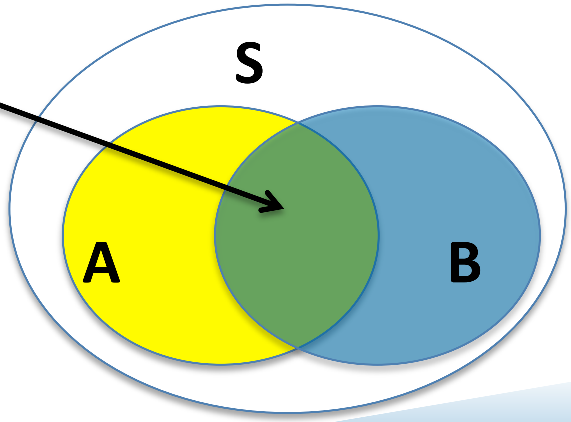
- Given an experiment, a corresponding sample space S, and a probability law
- Suppose we know that the outcome is within some given event B
- We want to quantify the likelihood that the outcome also belongs to some other given event A.
- We need a new probability law that gives us the conditional probability of A given B P(A|B)

Conditional Probability

- Back to cards:
  - P(king) = 4 / 52 = 1/13
  - P(red) = 26 / 52 = 1/2
- What is the probability that a card is a king if I know that it is red?
  - P(king | red) = $\dfrac{P(king \cap red)}{P(red)} = \dfrac{1/26}{26/52} = \dfrac{1}{13}$

Conditional Probability

- Let A and B be events
- $p(B|A)$ = the probability of event B occurring given event A occurs
- Definition: $p(B|A) = p(A \cap B) / p(A)$
- Notation & Notes
  - $p(A,B) = p(A \cap B)$
  - $p(A,B) = P(B,A)$
  - $p(A|B) = p(A \cap B) / p(B)$
  - $p(A,B) = p(A|B) * P(B)$



Independence

- What is P(A,B) if A and B are independent?
- P(A,B)=P(A) · P(B) iff A,B independent.
  - P(heads,tails) = P(heads) · P(tails) = .5 · .5 = .25
  - *Note: P(A|B)=P(A) iff A,B independent*
  - *Also: P(B|A)=P(B) iff A,B independent*

An Example

- What's the probability of a patient with a stiff neck having meningitis?
  - Meningitis causes a stiff neck in 50% of cases
    - p(S|M) = 1/2
  - The probability of having meningitis
    - p(M) = 1/50,000
  - The probability of having a stiff neck
    - p(S) = 1/20
  - $P(M|S) = \dfrac{P(S|M)P(M)}{P(S)} = \dfrac{½ * 1/50{,}000}{1/20} = 1/5000$

From Artificial Intelligence, Russel & Norvig, Prentice Hall, 1999

Bayes Theorem

$$P(B \mid A) = \frac{P(A|B)P(B)}{P(A)}$$

- Swap the conditioning
- Sometimes it's easier to estimate one kind of dependence than another

Language Modeling

- Back to word prediction
- We can model the word prediction task as the ability to assess the conditional probability of a word given the previous words in the sequence
  - $P(w_n | w_1, w_2 \ldots w_{n-1})$
- We'll call a statistical model that can assess this a *Language Model*

Language Modeling

- How might we go about calculating such a conditional probability?
  - One way is to use the definition of conditional probabilities and look for counts. So to get
  - P(*the* | *its water is so transparent that*)
- By definition that's

$$\frac{P(\text{its water is so transparent that the})}{P(\text{its water is so transparent that})}$$

We can get each of those from counts in a large corpus.

Very Easy Estimate

- How to estimate?
  - P(the | its water is so transparent that)

$$P(\text{the} \mid \text{its water is so transparent that}) = \frac{\text{Count(its water is so transparent that the)}}{\text{Count(its water is so transparent that)}}$$

Very Easy Estimate

- According to Google those counts are 5/9.
  - Unfortunately... 2 of those were to these slides... So maybe it's really
  - 3/7
  - In any case, that's not terribly convincing due to the small numbers involved.

(actually, it's 11,900 / 17,900 or .66)

Language Modeling

- Unfortunately, for most sequences and for most text collections we won’t get good estimates from this method.
  - What we’re likely to get is 0. Or worse 0/0.
- Clearly, we’ll have to be a little more clever.
  - Let’s use the chain rule of probability
  - And a particularly useful independence assumption.

The Chain Rule

- Recall the definition of conditional probabilities

$$P(A \mid B) = \frac{P(A \wedge B)}{P(B)}$$

- Rewriting:

$$P(A \wedge B) = P(A \mid B)P(B)$$

- For sequences...
  - $P(A,B,C,D) = P(A)P(B|A)P(C|A,B)P(D|A,B,C)$
- In general
  - $P(x_1,x_2,x_3,\ldots x_n) = P(x_1)P(x_2|x_1)P(x_3|x_1,x_2)\ldots P(x_n|x_1\ldots x_{n-1})$

The Chain Rule

$$
\begin{aligned}
P(w_1^n) &= P(w_1)P(w_2|w_1)P(w_3|w_1^2)\ldots P(w_n|w_1^{n-1}) \\
&= \prod_{k=1}^{n} P(w_k|w_1^{k-1})
\end{aligned}
$$

P(its water was so transparent)=

P(its)*

P(water|its)*

P(was|its water)*

P(so|its water was)*

P(transparent|its water was so)

Unfortunately

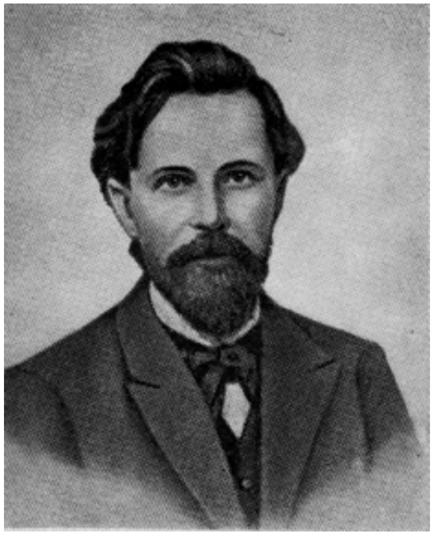
- There are still a lot of possible sentences
- In general, we'll never be able to get enough data to compute the statistics for those longer prefixes
  - – Same problem we had for the strings themselves

Independence Assumption

- Make the simplifying assumption
  - P(lizard|the,other,day,I,was,walking,along,and,saw,a) = P(lizard|a)
- Or maybe
  - P(lizard|the,other,day,I,was,walking,along,and,saw,a) = P(lizard|saw,a)
- That is, the probability in question is independent of its earlier history.

Independence Assumption

- This particular kind of independence assumption is called a *Markov assumption* after the Russian mathematician Andrei Markov.



Markov Assumption

So for each component in the product replace with the approximation (assuming a prefix of N)

$$P(w_n \mid w_1^{n-1}) \approx P(w_n \mid w_{n-N+1}^{n-1})$$

Bigram version

$$P(w_n \mid w_1^{n-1}) \approx P(w_n \mid w_{n-1})$$

Estimating Bigram Probabilities

- The Maximum Likelihood Estimate (MLE)

$$P(w_i \mid w_{i-1}) = \frac{count(w_{i-1}, w_i)}{count(w_{i-1})}$$

An Example

- <s> I am Sam </s>
- <s> Sam I am </s>
- <s> I do not like green eggs and ham </s>

$$P(\texttt{I} \mid \texttt{<s>}) = \tfrac{2}{3} = .67 \qquad P(\texttt{Sam} \mid \texttt{<s>}) = \tfrac{1}{3} = .33 \qquad P(\texttt{am} \mid \texttt{I}) = \tfrac{2}{3} = .67$$

$$P(\texttt{</s>} \mid \texttt{Sam}) = \tfrac{1}{2} = 0.5 \qquad P(\texttt{Sam} \mid \texttt{am}) = \tfrac{1}{2} = .5 \qquad P(\texttt{do} \mid \texttt{I}) = \tfrac{1}{3} = .33$$

$$P(w_n|w_{n-N+1}^{n-1}) = \frac{C(w_{n-N+1}^{n-1}w_n)}{C(w_{n-N+1}^{n-1})}$$

Maximum Likelihood Estimates

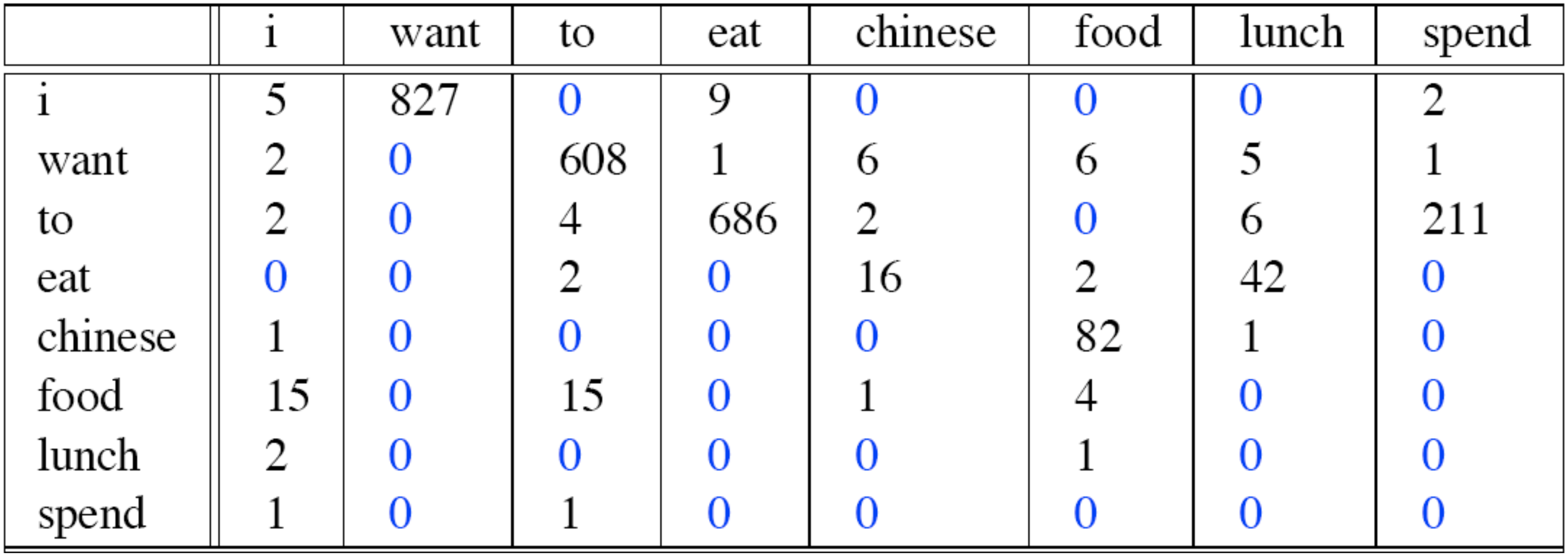
- The maximum likelihood estimate of some parameter of a model M from a training set T
  - Is the estimate that maximizes the likelihood of the training set T given the model M
- Suppose the word Chinese occurs 400 times in a corpus of a million words (Brown corpus)
- What is the probability that a random word from some other text from the same distribution will be "Chinese"
- MLE estimate is 400/1000000 = .004
  - This may be a bad estimate for some other corpus
- But it is the **estimate** that makes it **most likely** that "Chinese" will occur 400 times in a million word corpus.

Berkeley Restaurant Project Sentences

- *can you tell me about any good cantonese restaurants close by*
- *mid priced thai food is what i'm looking for*
- *tell me about chez panisse*
- *can you give me a listing of the kinds of food that are available*
- *i'm looking for a good place to eat breakfast*
- *when is caffe venezia open during the day*

Bigram Counts

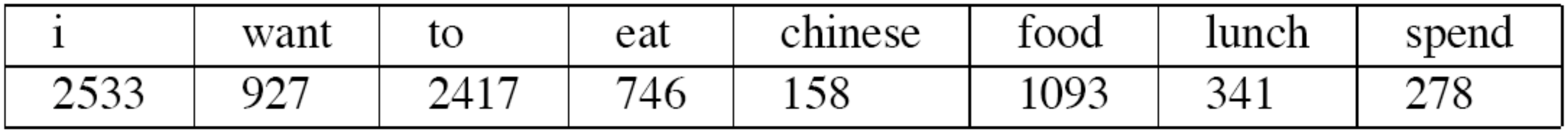
- Out of 9222 sentences
  - Eg. “I want” occurred 827 times



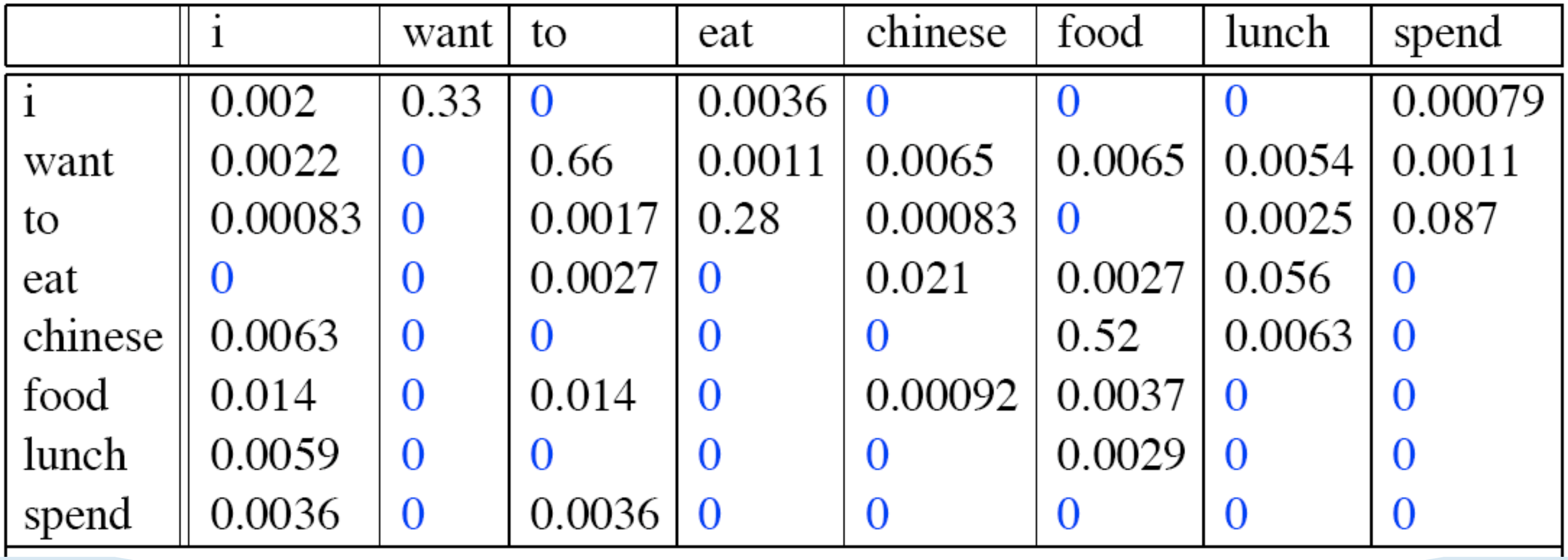
	i	want	to	eat	chinese	food	lunch	spend
i	5	827	0	9	0	0	0	2
want	2	0	608	1	6	6	5	1
to	2	0	4	686	2	0	6	211
eat	0	0	2	0	16	2	42	0
chinese	1	0	0	0	0	82	1	0
food	15	0	15	0	1	4	0	0
lunch	2	0	0	0	0	1	0	0
spend	1	0	1	0	0	0	0	0

Bigram Probabilities

- Divide bigram counts by prefix unigram counts



i	want	to	eat	chinese	food	lunch	spend
2533	927	2417	746	158	1093	341	278



	i	want	to	eat	chinese	food	lunch	spend
i	0.002	0.33	0	0.0036	0	0	0	0.00079
want	0.0022	0	0.66	0.0011	0.0065	0.0065	0.0054	0.0011
to	0.00083	0	0.0017	0.28	0.00083	0	0.0025	0.087
eat	0	0	0.0027	0	0.021	0.0027	0.056	0
chinese	0.0063	0	0	0	0	0.52	0.0063	0
food	0.014	0	0.014	0	0.00092	0.0037	0	0
lunch	0.0059	0	0	0	0	0.0029	0	0
spend	0.0036	0	0.0036	0	0	0	0	0

Bigram Estimates of Sentence Probabilities

- P(<s> I want english food </s>) =

  P(i|<s>)*

  P(want|I)*

  P(english|want)*

  P(food|english)*

  P(</s>|food)*

  =.000031

Kinds of Knowledge

- As crude as they are, *N*-gram probabilities capture a range of interesting facts about language.

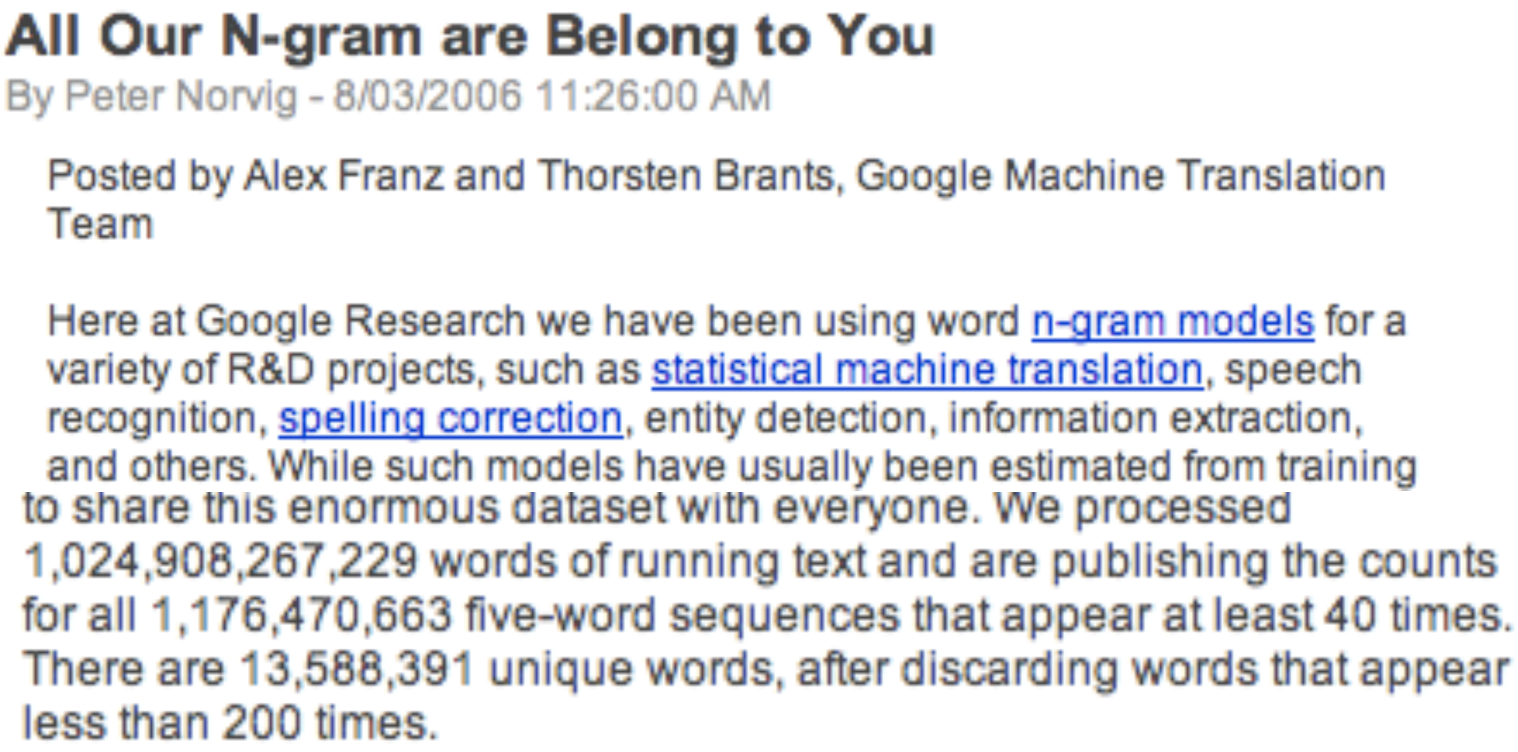
- P(english|want) = .0011
- P(chinese|want) = .0065
- P(to|want) = .66
- P(eat | to) = .28
- P(food | to) = 0
- P(want | spend) = 0
- P (i | <s>) = .25

World knowledge

Syntax

Discourse

Google N-Gram Release



All Our N-gram are Belong to You

By Peter Norvig - 8/03/2006 11:26:00 AM

Posted by Alex Franz and Thorsten Brants, Google Machine Translation Team

Here at Google Research we have been using word n-gram models for a variety of R&D projects, such as statistical machine translation, speech recognition, spelling correction, entity detection, information extraction, and others. While such models have usually been estimated from training

to share this enormous dataset with everyone. We processed 1,024,908,267,229 words of running text and are publishing the counts for all 1,176,470,663 five-word sequences that appear at least 40 times. There are 13,588,391 unique words, after discarding words that appear less than 200 times.

Available through LDC:

http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2006T13

Google N-Gram Release

- serve as the incoming 92
- serve as the incubator 99
- serve as the independent 794
- serve as the index 223
- serve as the indication 72
- serve as the indicator 120
- serve as the indicators 45
- serve as the indispensable 111
- serve as the indispensible 40
- serve as the individual 234

Google 4-grams

- Highest
  - serve as the initial 5331
  - serve as the inspiration 1390
  - serve as the input 1323
  - serve as the information 838
  - serve as the independent 794
- Lowest
  - serve as the informational 41
  - serve as the inlet 41
  - serve as the indispensible 40

Google Caveat

- Remember the lesson about test sets and training sets... Test sets should be similar to the training set (drawn from the same distribution) for the probabilities to be meaningful.
- So... The Google corpus is fine if your application deals with arbitrary English text on the Web.
- If not then a smaller domain specific corpus is likely to yield better results.