Corpus Statistics: Unigram/Word Frequencies

School of Information Studies
Syracuse University

What Is Corpus Statistics/Linguistics?

A methodology to process text and provide information about the text

The corpus is a collection of text

- Utilizes a representative sample of machine-readable text of a language or a particular variety of text or language

Statistical analysis

- Word frequencies, referred to as unigram frequencies
- Collocations of words: bigrams, trigrams, etc.

Often used in "digital humanities" as ways to characterize properties of corpora

- Where the "properties" of interest may govern choices of words to highlight

Word Frequencies

Count the number of each token appearing in the corpus (or sometimes single document).

A frequency distribution is a list of all tokens with their frequency, usually sorted in the order of decreasing frequency.

They are used to make "word clouds," where the most frequent words will be made bigger.

How Many Words in a Corpus?

Let N be the number of tokens (words plus other symbols).

Let V be the size of the vocabulary (the number of distinct tokens).

	Tokens = N	Types = $\|V\|$
Switchboard phone conversations	2.4 million	20 thousand
Shakespeare	884,000	31 thousand
Google N-grams	1 trillion	13 million

from Dan Jurafsky

Also see xkcd.com/1133/
How to describe rocket only using words from most common 1,000

Zipf's Law

In a natural language corpus, the frequency of any word is inversely proportional to its rank in a frequency table.

Rank (r): The numerical position of a word in a list sorted by decreasing frequency (f).

Zipf (1949) "discovered" that: $f \ r = k$ (for constant k)

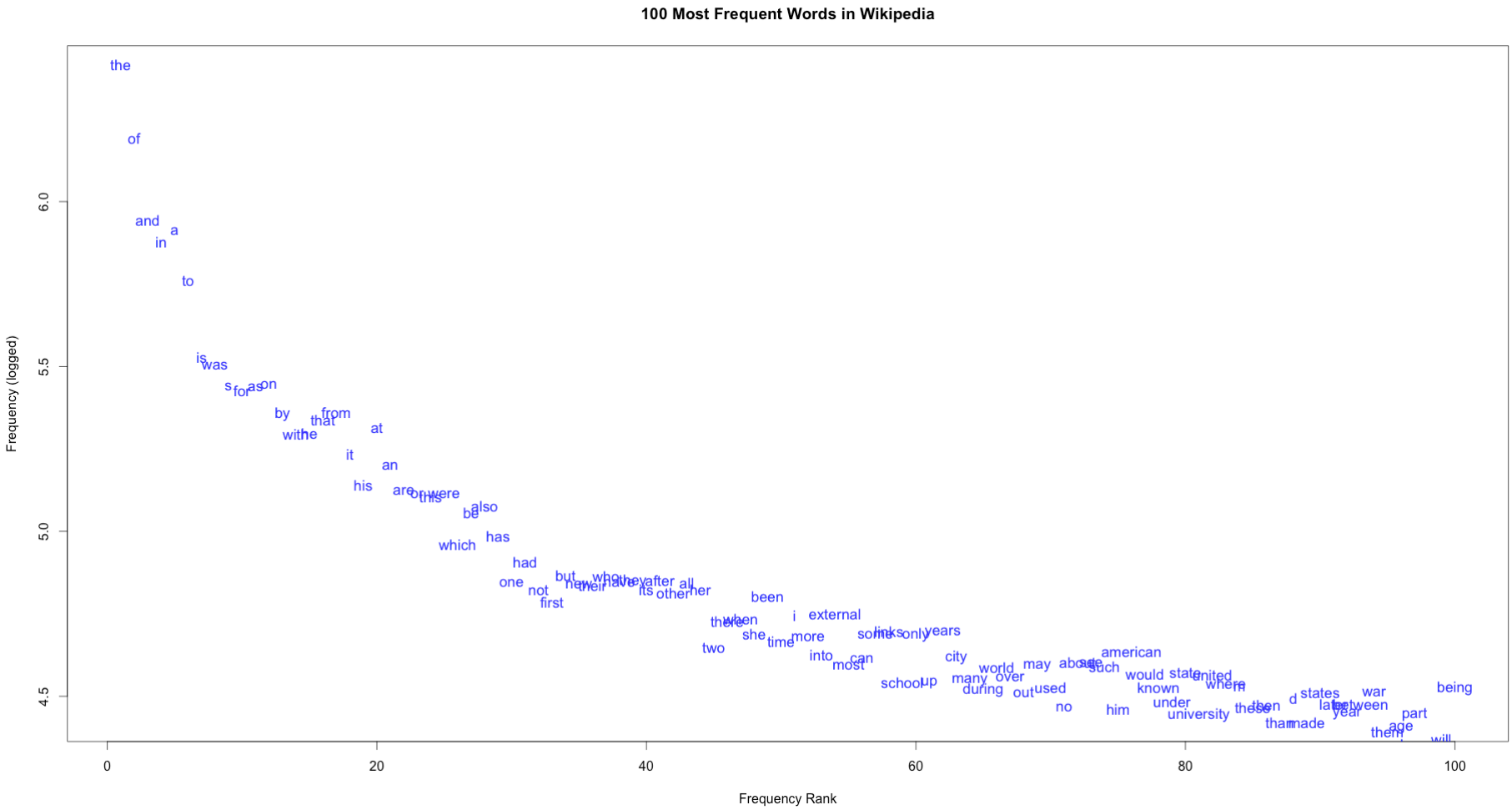
- Examples if k is 1:
  - Most frequent word ($r = 1$) is twice as frequent as second most frequent
  - Most frequent ($r = 1$) is three times as frequent as third most frequent, etc.

Zipf's Law

For example, in the Brown Corpus of American English text,

- the word "the" is the most frequently occurring word and, by itself, accounts for nearly 7% of all word occurrences (69,971 out of slightly over 1 million),
- the second-place word "of" accounts for slightly over 3.5% of the words (36,411 occurrences),
- followed by "and" (28,852).

—from Wikipedia



A sample of 36.8 million words from Wikipedia, over 580,000 word types, nearly half (280,000) occur just once in the sample (image and this data from http://wugology.com/zipfs-law/)

Zipf's Law Impact on Language Analysis

Good news: Stopwords (commonly occurring words such as "the") will account for a large fraction of text, so eliminating them greatly reduces the number of words in a text.

Bad news: For most words, gathering sufficient data for meaningful statistical analysis is difficult since they are extremely rare.

Corpus Statistics: Bigram Frequencies and Mutual Information (MI)

School of Information Studies
Syracuse University

Bigrams

Examples of bigrams are any two words that occur together in the text.

- In "two great and powerful groups of nations,"
- the bigrams are "two great," "great and," "and powerful," etc.

The *frequency* of an *N*-gram is the percentage of times the *N*-gram occurs in all the *N*-grams of the corpus and could be useful in corpus statistics.

- For bigram *xy*:
  - Count of bigram *xy*/count of all bigrams in the corpus
- Examples are in the Google *N*-gram corpus

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.

Example Data

Examples of 4-gram frequencies from the Google *N*-gram release

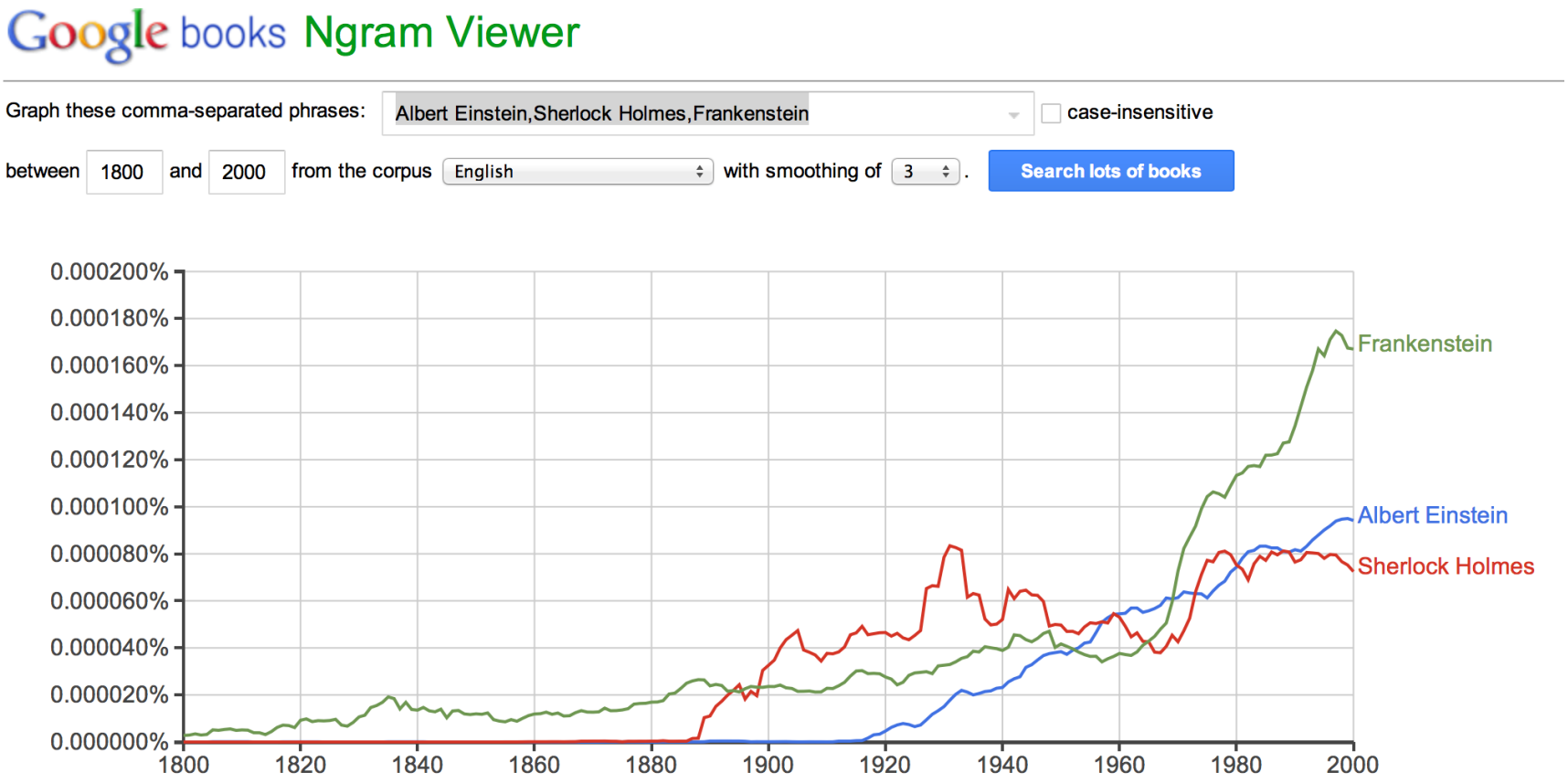
- `serve as the incoming 92`
- `serve as the incubator 99`
- `serve as the independent 794`
- `serve as the index 223`
- `serve as the indication 72`
- `serve as the indicator 120`
- `serve as the indicators 45`
- `serve as the indispensable 111`
- `serve as the indispensable 40`
- `serve as the individual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

Google Ngram Viewer

In 2010, Google placed online the Ngram Viewer that would display graphs of *N*-gram frequencies of one or more *N*-grams, based on a corpus defined from Google Books.

- https://books.google.com/ngrams
- And see also the "About NGram Viewer" link

Google Ngram Viewer



Additional Corpus Measures

Recall that, so far, we have looked at one measure involving bigrams (and these definitions can be extended to *N*-grams).

- Bigram frequency: percentage occurrence of the bigram in the corpus
  - Seen in the Google Ngram data

Other measures can be defined about the occurrences of bigrams in a corpus.

- Mutual information, …
- More of these can be found in the NLTK

Corpus Statistics: Mutual Information (MI)

Mutual information computes the probability of two words occurring in sequence.

Given a pair of words, it compares the probability that the two occur together as a joint event with the probability that they occur individually and that their co-occurrences are simply the result of chance.

- The more strongly connected two items are, the higher will be their MI value.

Mutual Information

Based on the work of Church and Hanks (1990), generalizing MI from information theory to apply to words in sequence

- They used the terminology *association ratio*

$P(x)$ and $P(y)$ are estimated by the number of observations of x and y in a corpus and normalized by N, the size of the corpus

$P(x,y)$ is the number of times that x is followed by y in a window of w words

Mutual information score (also sometimes called PMI, pointwise mutual information):

$$\text{PMI}(x,y) = \log_2 (P(x,y)/P(x)P(y))$$

MI Values Based on 145 *Wall Street Journal* Articles

x	freq (*x*)	*y*	freq (*y*)	freq (*x*,*y*)	MI
Gaza	3	Strip	3	3	14.42
joint	8	venture	4	4	13.00
Chapter	3	11	14	3	12.20
credit	15	card	11	7	11.44
average	22	yield	7	5	11.06
appeals	4	court	47	4	10.45
.....					
said	444	it	346	76	5.02

Uses of Mutual Information

Can be used to characterize text in corpus statistics, but also in other NLP processes

- Idiomatic phrases for machine translation
- Sense disambiguation (both statistical and symbolic approaches)
- Error detection and correction in speech analysis and spell-checking

Used for distributional semantics in “deep learning”

Used in non-text applications such as comparing features in machine learning

Corpus Statistics: Examples

School of Information Studies
Syracuse University

Characterizing Text

One method of digital humanities is to characterize text with corpus statistics

- Collect frequencies of words and bigrams or mutual information scores

Use these to characterize some aspect of the text, often used to compare two or more texts

Examples of characteristics

- Contents, topics
- Style, informal vs. formal, differences in gender usage

Example 1: SOTU Speeches

Example of word frequencies for comparison and characterization of text

See the State of the Union (SOTU) Speeches by Nate Silver
http://fivethirtyeight.com/features/obamas-sotu-clintonian-in-good-way/

Comparison questions:

- How do the topics in the SOTU speeches of one president compare with another's?
- In the article, several different comparison questions are discussed.

Methodology: choose topic words of interest and plot frequencies of these words vs. different speeches

Example 2: Potato Chip Ad Language

A student in an NLP class with Dan Jurafsky collected text from regular and gourmet potato chip bags.

https://web.stanford.edu/~jurafsky/freedmanjurafsky2011.pdf

"Our goal, then, is to explore whether advertising on chips targeted toward consumers of high socioeconomic status uses different language than that on chips designed to appeal to lower status consumers. We hope to better understand how advertisers distinguish the concepts of food for the upper class and the working class or lower-middle class in America."

Methodology: The student looked at the complexity of the text, using a specific measure based on word frequencies.

Language Models/ *N*-Gram Models

School of Information Studies
Syracuse University

Language Models

The goal of a language model is to assign a probability that a sentence (or phrase) will occur in natural uses of the language.

Why?

- Machine translation
  - *P*(**high** winds tonight) > *P*(**large** winds tonight)
- Spell correction
  - The office is about 15 **minuets** from my house
    - *P*(about 15 **minutes** from) > *P*(about 15 **minuets** from)
- Speech recognition
  - *P*(I saw a van) >> *P*(eyes awe of an)
- And other NLP applications

Corpus statistics captures a static view of a corpus, but language models use a corpus to predict how words occur in sentences.

Language Models

Goal: compute the probability of a sentence or sequence of words

- $P(W) = P(w_1,w_2,w_3,w_4,w_5 \ldots w_n)$

Related task: probability of an upcoming word

- $P(w_5|w_1,w_2,w_3,w_4)$
- Conditional probability that w_5 occurs, given that we know that w_1,w_2,w_3,w_4 already occurred

A model that computes either of these:

- $P(W)$ or $P(w_n|w_1,w_2 \ldots w_{n-1})$ is called a **language model**

We might call this a grammar because it predicts the (word-level) structure of the language, but language model is the standard terminology

Chain Rule Applied

Compute the probability of a sentence by computing the joint probability of all the words conditioned by the previous words.

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

Example: P("its water is so transparent") =

P(its) × P(water|its) × P(is|its water) × P(so|its water is) × P(transparent|its water is so)

With our corpus, we can just compute the probability that something occurred by counting its occurrences and dividing by the total number.

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

But there are way too many unique English sentences in any realistic corpus for this to work! We'll never see enough data.

Markov Assumption

Instead, we make the simplifying Markov assumption that we can predict the next word based on only one word previous:

Andrei Markov

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

Or perhaps two words previous:

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

N-Gram Models

Unigram model(word frequencies): The simplest case is that we predict a sentence probability just based on the probabilities of the words with no preceding words.

$$P(w_1 w_2 \ldots w_n) \approx \prod_i P(w_i)$$

Bigram model (two word frequencies): Prediction is based on one previous word:

$$P(w_i \mid w_1 w_2 \ldots w_{i-1}) \approx P(w_i \mid w_{i-1})$$

N-Gram Models

We can extend to trigrams, 4-grams, and 5-grams.

- Each higher number will get a more accurate model, but it will be harder to find examples of the longer word sequences in the corpus

In general, this is an insufficient model of language because:

- Language has **long-distance dependencies**
  - *"The computer that I had just put into the machine room on the fifth floor crashed."*
- The last word *crashed* is not very likely to follow the word *floor*, but it is likely to be the main verb of the word *computer*

But we can often get away with *N*-gram models.

N-Gram Predictive Probabilities

For *N*-grams, we need the conditional probability:

P(<next word>|<preceding word sequence of length *n*>)
(e.g., *P*(*the*|*They picnicked by*))

We define this as:

- The observed frequency (count) of the whole sequence divided by
- The observed frequency of the preceding, or initial, sequence (sometimes called the maximum likelihood estimation (MLE):

  P(<next word>|<preceding word sequence of length *n*>)
  = Count (<preceding word sequence> <next word>)
  / Count (<preceding word sequence>)

- Example: Count (*They picnicked by the*) / Count (*They picnicked by*)

Language Modeling Examples

School of Information Studies
Syracuse University

Example of Bigram Predictive Probabilities

Divide the count of the bigram by the count of the first word:

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

Example mini-corpus of three sentences, where we have separated sentences, and we include the sentence tags in order to represent the beginning and end of the sentence:

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

Bigram predictive probabilities:

P (I | <S>) = 2/3 = .67 (probability that I follows <S>)
P (</S> | Sam) = 1/2 = .5
P (Sam | <S>) = 1/3 = .33
P (Sam | am) = 1/2 = .5
P (am | I) = 2/3 = .67

Example Using Bigram Predictive Probabilities to Predict the Probabilities of Sentences

Berkeley Restaurant Project collected online questions from people about restaurants in the Berkeley area.

Some example 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*

Raw Bigram Counts From the Corpus

Out of 9,222 sentences, showing counts that the word on the left is followed by the word on the top

	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 Predictive Probabilities

Unigram counts:

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

Bigram Predictive Probabilities

Resulting bigram predictive probability
(bigram counts/unigram counts of first words):

	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

Using *N*-Grams for Sentences

For a bigram grammar: $\prod_{k=1}^{n} P(w_k \mid w_{k-1})$

- *P*(sentence) can be approximated by multiplying all the bigram probabilities in the sequence

Example of using bigram probabilities to compute the probability of a sentence:

P(I want to eat Chinese food) =

P(I|<S>) *P*(want|I) *P*(to|want) *P*(eat|to) *P*(Chinese|eat) *P*(food|Chinese)

More Bigrams From the Restaurant Corpus

Eat on	.16	Eat Thai	.03
Eat some	.06	Eat breakfast	.03
Eat lunch	.06	Eat in	.02
Eat dinner	.05	Eat Chinese	.02
Eat at	.04	Eat Mexican	.02
Eat a	.04	Eat tomorrow	.01
Eat Indian	.04	Eat dessert	.007
Eat today	.03	Eat British	.001

Examples due to Rada Mihalcea

Additional Bigrams

<S> I	.25	Want some	.04
<S> I'd	.06	Want Thai	.01
<S> Tell	.04	To eat	.26
<S> I'm	.02	To have	.14
I want	.32	To spend	.09
I would	.29	To be	.02
I don't	.08	British food	.60
I have	.04	British restaurant	.15
Want to	.65	British cuisine	.01
Want a	.05	British lunch	.01

Computing Sentence Probabilities

P (I want to eat British food) = P (I | <S>) P (want | I) P (to | want) P (eat | to) P (British | eat) P (food | British) = .25 × .32 × .65 × .26 × .001 × .60 = .000080

Vs.

P (I want to eat Chinese food) = .00015

Probabilities seem to capture "syntactic" facts, "world knowledge"

- *Eat* is often followed by a noun
- British food is not too popular

N-gram models can be trained by counting and normalization

Language Modeling Smoothing

School of Information Studies
Syracuse University

Using *N*-Gram Probabilities in a Language Model: Why Do We Need Smoothing?

Every *N*-gram training matrix is sparse, even for very large corpora (remember Zipf's law).

- There are words that don't occur in the training corpus that may occur in future text.
- These are known as the unseen words.

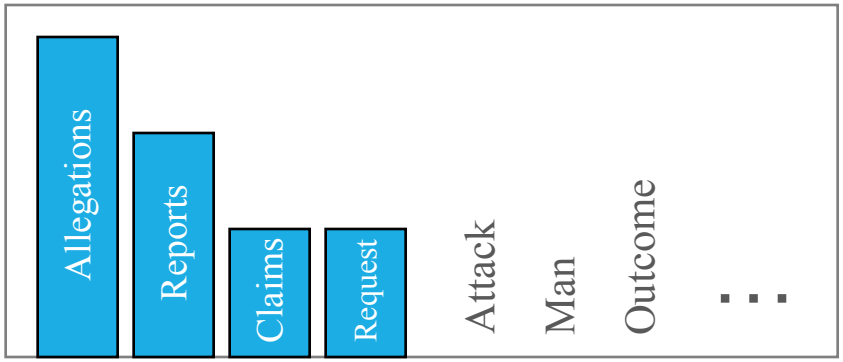
Whenever a probability is 0, it will multiply the entire sequence to be 0.

Solution: Estimate the likelihood of unseen *N*-grams, and include a small probability for unseen words.

Intuition of Smoothing
(From Dan Klein)

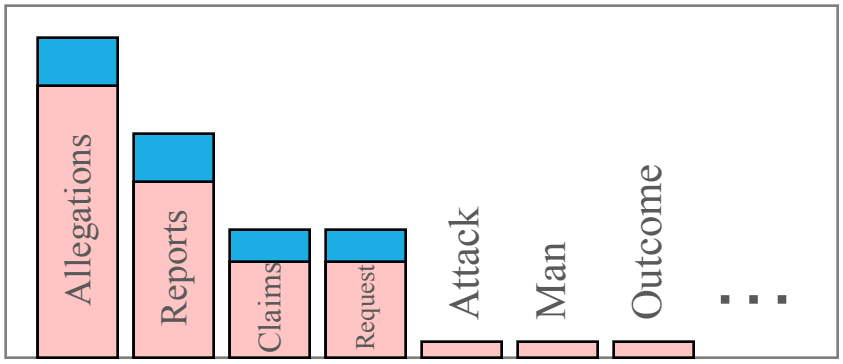
When we have sparse statistics:

P(*w*|denied the)
3 allegations
2 reports
1 claims
1 request
7 total



Steal probability mass to generalize better:

P(*w*|denied the)
2.5 allegations
1.5 reports
0.5 claims
0.5 request
2 other
7 total



Smoothing

Add-one smoothing

- Given: $P(w_n|w_{n-1}) = C(w_{n-1}w_n)/C(w_{n-1})$
- Add 1 to each count: $P(w_n|w_{n-1}) = [C(w_{n-1}w_n) + 1]/[C(w_{n-1}) + V]$

Back-off smoothing for higher-order *N*-grams

- Notice that:
  - *N*-grams are more precise than (*N*–1) grams
  - But also, *N*-grams are sparser than (*N*–1) grams
- How to combine things?
  - Attempt *N*-grams and back off to (*N*–1) if counts are not available
  - Example: attempt prediction using 4-grams, and back off to trigrams (or bigrams, or unigrams) if counts are not available

More complicated techniques exist: in practice, NLP language models use Knesser-Ney smoothing

Language-Modeling Toolkit

Language-modeling tools help you define your language models by making those tables and calculating the probabilities.

SRI language modeling:

- http://www.speech.sri.com/projects/srilm/
- http://www.speech.sri.com/projects/srilm/papers/icslp2002-srilm.pdf
  - A conference paper that gives an overview of the toolkit