n-grams

BM1: Advanced Natural Language Processing

University of Potsdam

Tatjana Scheffler

tatjana.scheffler@uni-potsdam.de

October 28, 2016

Today

- n-grams
- Zipf's law
- language models

Maximum Likelihood Estimation

- We want to estimate the parameters of our model from frequency observations. There are many ways to do this. For now, we focus on *maximum likelihood estimation*, MLE.
- *Likelihood* L(O ; p) is the probability of our model generating the observations O, given parameter values p.
- Goal: Find value for parameters that maximizes the likelihood.

Bernoulli model

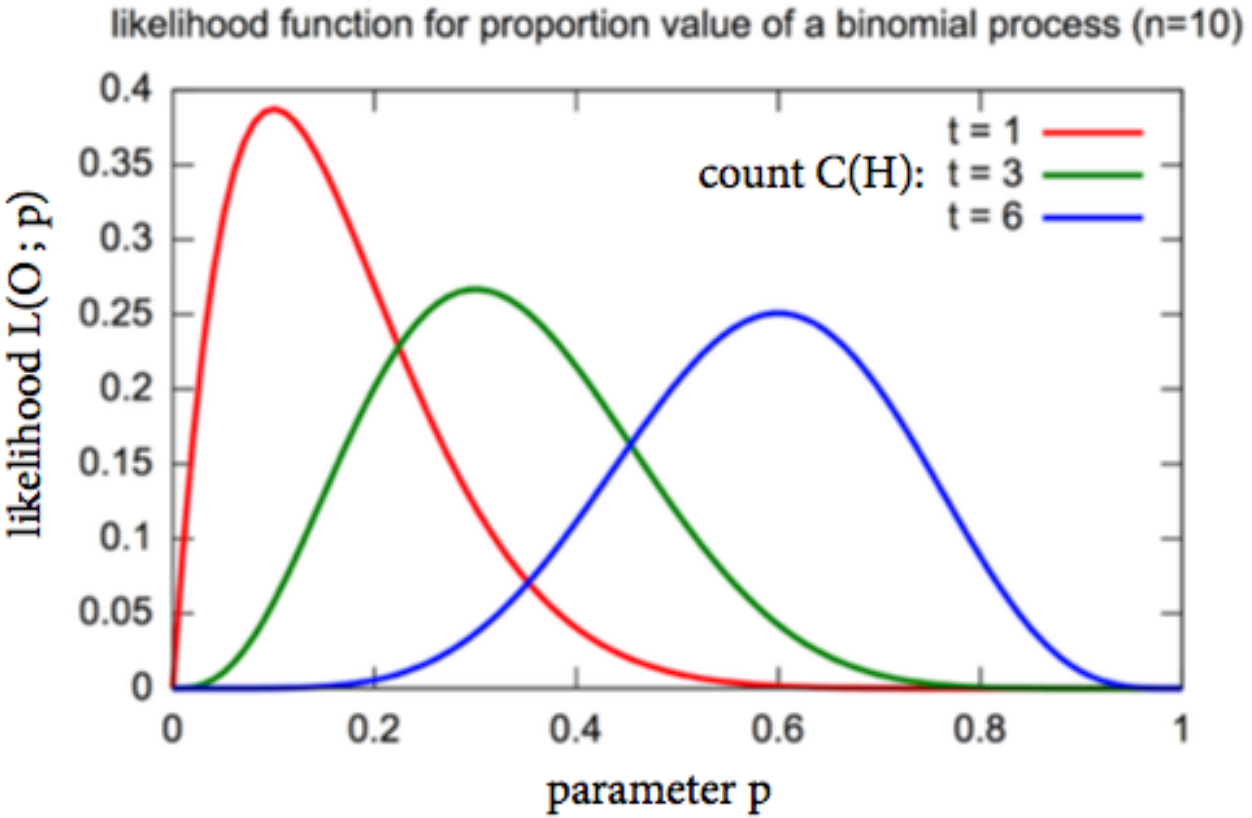
- Let's say we had training data C of size N, and we had N_H observations of H and N_T observations of T.

likelihood $$L(C) = \prod_{i=1}^{N} P(w_i \mid p) = \prod_{i=1}^{N} p^{N_H}(1-p)^{N_T}$$

log-likelihood

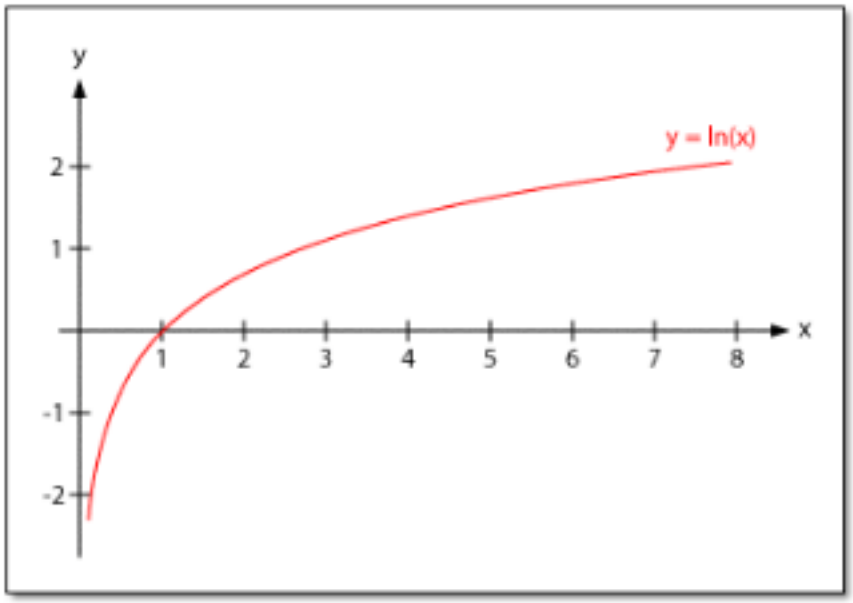
$$\ell(C) = \log L(C) = \sum_{i=1}^{N} \log P(w_i \mid p) = N_H \log p + N_T \log(1-p)$$

Likelihood functions



(Wikipedia page on MLE; licensed from Casp11 under CC BY-SA 3.0)

Logarithm is monotonic



- Observation: If $x_1 > x_2$, then $\ln(x_1) > \ln(x_2)$.
- Therefore, argmax L(C) = argmax l(C)
  p p

Maximizing the log-likelihood

- Find maximum of function by setting derivative to zero:

$$\ell(C) = N_H \log p + N_T \log(1-p)$$

$$\frac{d\ell(C)}{dp} = \frac{N_H}{p} - \frac{N_T}{1-p}$$

- Solution is $p = N_H / N = f(H)$.

Language Modelling

Let's play a game

- I will write a sentence on the board.
- Each of you, in turn, gives me a word to continue that sentence, and I will write it down.

Let's play another game

- You write a word on a piece of paper.
- You get to see the piece of paper of your neighbor, but none of the earlier words.
- In the end, I will read the sentence you wrote.

Statistical models for NLP

- *Generative statistical model* of language:

  prob. dist. P(w) over NL expressions that we can observe.
  - w may be complete sentences or smaller units
  - will later extend this to pd P(w, t) with hidden random variables t

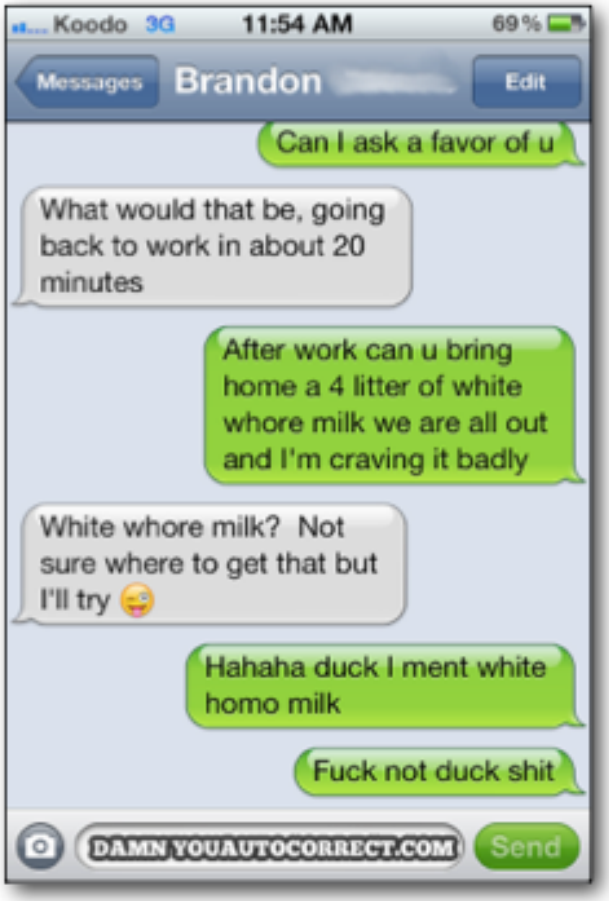
- Assumption: A *corpus* of observed sentences w is generated by repeatedly sampling from P(w).

- We try to estimate the *parameters* of the prob dist from the corpus, so we can make *predictions* about unseen data.

Example


Fido 3G
9:40 PM
56%
Messages
Anna
Edit
2012-05-17 9:31 PM
Yeah come by
Ok gonna be there in 5 monsoons
So 5 years?
Delivered
DAMN YOUAUTOCORRECT.COM
Send


Orange...
12:28
87%
Messages
Tinkler
Edit
:) . What are your plans for today then?x
Nothing beyond seeing you dead xx
DEAR!!!
DAMN YOUAUTOCORRECT.COM
Send


Koodo 3G
11:54 AM
69%
Messages
Brandon
Edit
Can I ask a favor of u
What would that be, going back to work in about 20 minutes
After work can u bring home a 4 litter of white whore milk we are all out and I'm craving it badly
White whore milk? Not sure where to get that but I'll try
Hahaha duck I ment white homo milk
Fuck not duck shit
DAMN YOUAUTOCORRECT.COM
Send

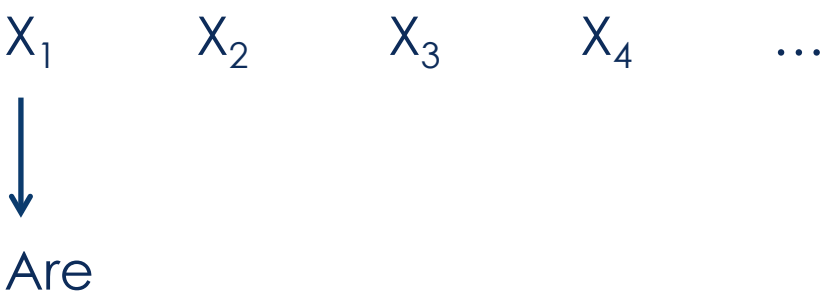
Word-by-word random process

- A *language model* LM is a probability distribution P(w) over words.
- Think of it as a random process that generates sentences word by word:

X_1 X_2 X_3 X_4 …

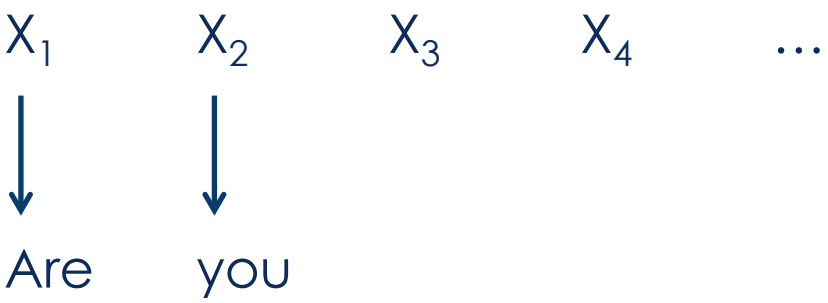
Word-by-word random process

- A *language model* LM is a probability distribution P(w) over words.
- Think of it as a random process that generates sentences word by word:



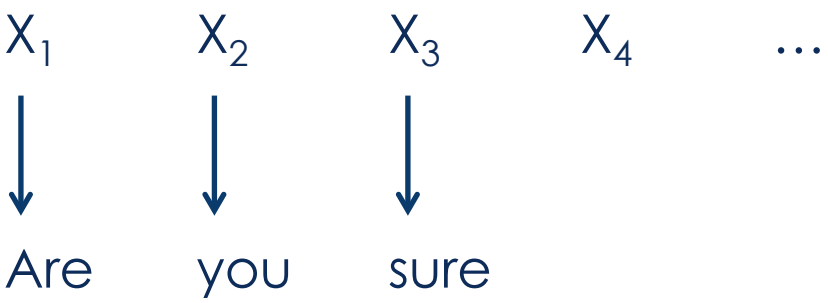
Word-by-word random process

- A *language model* LM is a probability distribution P(w) over words.
- Think of it as a random process that generates sentences word by word:



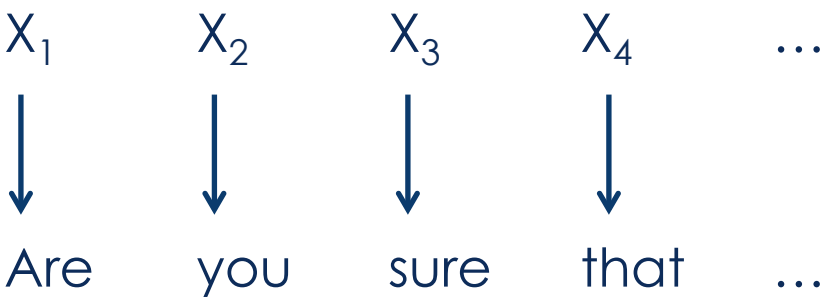
Word-by-word random process

- A *language model* LM is a probability distribution P(w) over words.
- Think of it as a random process that generates sentences word by word:



Word-by-word random process

- A *language model* LM is a probability distribution P(w) over words.
- Think of it as a random process that generates sentences word by word:



Our game as a process

- Each of you = a random variable X_t;

  event "$X_t = w_t$" means word at position t is w_t.

- When you chose w_t, you could see the outcomes of the previous variables: $X_1 = w_1, \ldots, X_{t-1} = w_{t-1}$.

- Thus, each X_t followed a pd

$$P(X_t = w_t \mid X_1 = w_1, \ldots, X_{t-1} = w_{t-1})$$

Our game as a process

- Assume that X_t follows some given pd

$$P(X_t = w_t \mid X_1 = w_1 , \ldots , X_{t-1} = w_{t-1})$$

- Then probability of the entire sentence (or corpus) $w = w_1 \ldots w_n$ is

$$P(w_1 \ldots w_n) = P(w_1)P(w_2 \mid w_1)P(w_3 \mid w_1, w_2) \ldots P(w_n \mid w_1, \ldots , w_{n-1})$$

Parameters of the model

- Our model has one parameter for

  $P(X_t = w_t \mid w_1, ..., w_{t-1})$ for all t and $w_1, ..., w_t$.

- Can use maximum likelihood estimation:

$$P(w_t \mid w_1, \ldots, w_{t-1}) = \frac{C(w_1 \ldots w_{t-1} w_t)}{C(w_1 \ldots w_{t-1})}$$

- Let's say a natural language has 10^5 different words. How many tuples $w_1, ... w_t$ of length t?
  - t = 1: 10^5
  - t = 2: 10^{10} different contexts
  - t = 3: 10^{15}; etc.

Sparse data problem

- typical corpus size:
  - Brown corpus: 10^6 tokens
  - Gigaword corpus: 10^9 tokens
- Problem exacerbated by *Zipf 's Law:*
  - Order all words by their absolute frequency in corpus (rank 1 = most frequent word).
  - Then rank is inversely proportional to absolute frequency; i.e., most words are really rare.
  - Zipf's Law is very robust across languages and corpora.

Interlude: Corpora

Terminology

- N = corpus size; number of (word) *tokens*
- V = vocabulary; number of (word) *types*
- hapax legomenon = a word that appears exactly once in the corpus

An example corpus

Es war einmal ein Müller, der hatte drei Söhne, seine Mühle, einen Esel und einen Kater; die Söhne mußten mahlen, der Esel Getreide holen und Mehl forttragen, die Katze dagegen die Mäuse wegfangen. Als der Müller starb, teilten sich die drei Söhne in die Erbschaft: der älteste bekam die Mühle, der zweite den Esel, der dritte den Kater; weiter blieb nichts für ihn übrig. Da war er traurig und sprach zu sich selbst:

- Tokens: 86
- Types: 53

Frequency list

Es	1		die	5
war	2		mußten	1
einmal	1		mahlen	1
ein	1		Getreide	1
Müller	2		holen	1
,	8		Mehl	1
der	5		forttragen	1
hatte	1		Katze	1
drei	2		dagegen	1
Söhne	3		Mäuse	1
seine	1		wegfangen	1
Mühle	2		.	3
einen	2		Als	1
Esel	3		starb	1
und	3		teilten	1
Kater	2		sich	2
;	2			

Frequency list

Type	abs.	relativ
,	8	0,083
der	5	0,052
die	5	0,052
.	3	0,031
Esel	3	0,031
Söhne	3	0,031
und	3	0,031
:	2	0,021
;	2	0,021
den	2	0,021
drei	2	0,021
einen	2	0,021
Kater	2	0,021
Mühle	2	0,021
Müller	2	0,021
sich	2	0,021
war	2	0,021

Type	abs.	relativ
Als	1	0,0104
älteste	1	0,0104
bekam	1	0,0104
blieb	1	0,0104
Da	1	0,0104
dagegen	1	0,0104
dritte	1	0,0104
ein	1	0,0104
einmal	1	0,0104
er	1	0,0104
Erbschaft	1	0,0104
Es	1	0,0104
forttragen	1	0,0104
für	1	0,0104
Getreide	1	0,0104
hatte	1	0,0104
holen	1	0,0104
ihn	1	0,0104

Type	abs.	relativ
in	1	0,01
Katze	1	0,01
Mäuse	1	0,01
Mehl	1	0,01
mußten	1	0,01
mahlen	1	0,01
nichts	1	0,01
seine	1	0,01
selbst	1	0,01
sprach	1	0,01
starb	1	0,01
teilten	1	0,01
traurig	1	0,01
übrig	1	0,01
wegfangen	1	0,01
weiter	1	0,01
zu	1	0,01
zweite	1	0,01

Frequency profile

Type	Rang	abs.	relativ
,	1	8	0,083
der	3	5	0,052
die	3	5	0,052
.	7	3	0,031
Esel	7	3	0,031
Söhne	7	3	0,031
und	7	3	0,031
:	17	2	0,021
;	17	2	0,021
den	17	2	0,021
drei	17	2	0,021
einen	17	2	0,021
Kater	17	2	0,021
Mühle	17	2	0,021
Müller	17	2	0,021
sich	17	2	0,021
war	17	2	0,021

Type	Rang	abs.	relativ
Als	53	1	0,0104
älteste	53	1	0,0104
bekam	53	1	0,0104
blieb	53	1	0,0104
Da	53	1	0,0104
dagegen	53	1	0,0104
dritte	53	1	0,0104
ein	53	1	0,0104
einmal	53	1	0,0104
er	53	1	0,0104
Erbschaft	53	1	0,0104
Es	53	1	0,0104
forttragen	53	1	0,0104
für	53	1	0,0104
Getreide	53	1	0,0104
hatte	53	1	0,0104
holen	53	1	0,0104
ihn	53	1	0,0104

Type	Rang	abs.	relativ
in	53	1	0,01
Katze	53	1	0,01
Mäuse	53	1	0,01
Mehl	53	1	0,01
mußten	53	1	0,01
mahlen	53	1	0,01
nichts	53	1	0,01
seine	53	1	0,01
selbst	53	1	0,01
sprach	53	1	0,01
starb	53	1	0,01
teilten	53	1	0,01
traurig	53	1	0,01
übrig	53	1	0,01
wegfangen	53	1	0,01
weiter	53	1	0,01
zu	53	1	0,01
zweite	53	1	0,01

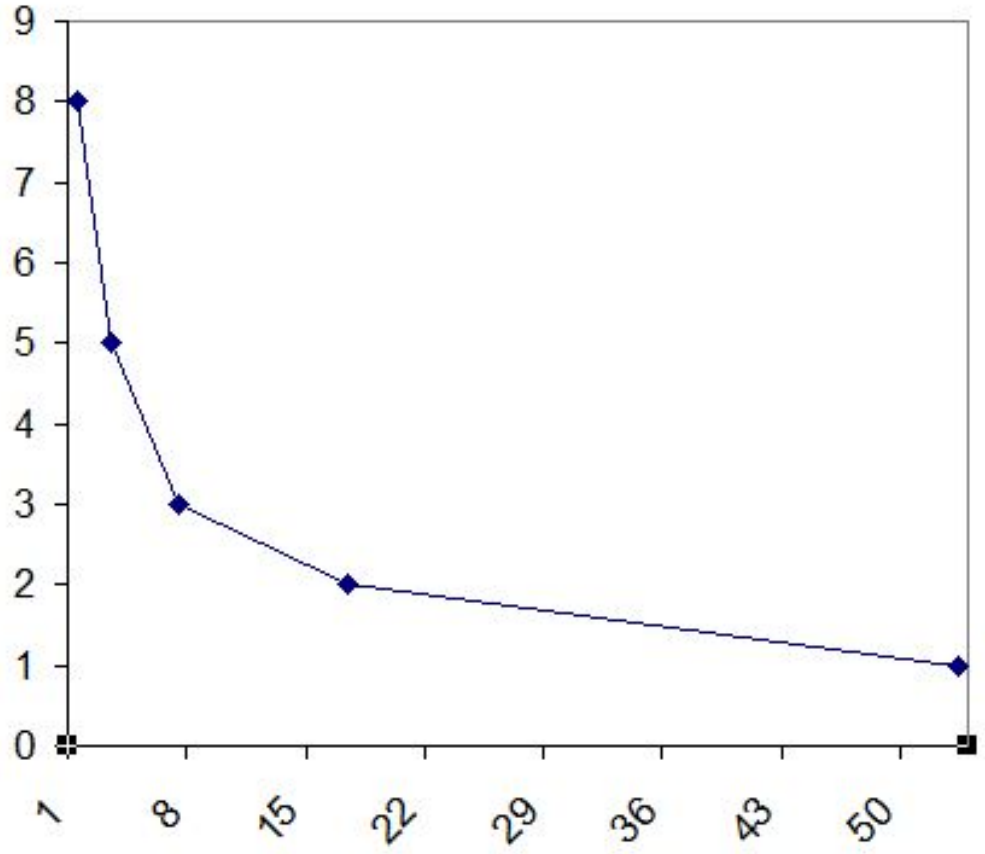
Plotting corpus frequencies

Number of types	rank	frequency
1	1	8
2	3	5
4	7	3
10	17	2
36	53	1

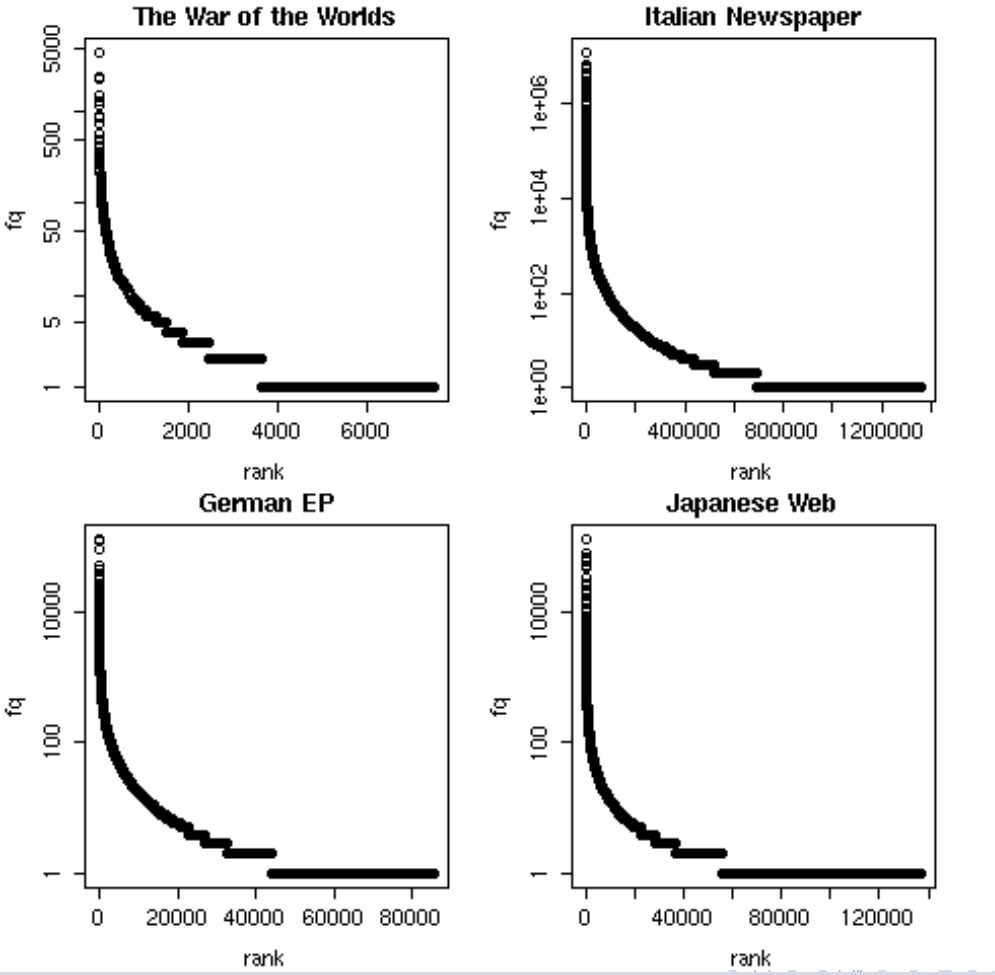
- How many different words in the corpus are there with each frequency?

Plotting corpus frequencies

- x-axis: rank
- y-axis: frequency



Some other corpora



Zipf's Law

- Zipf's Law characterizes the relation between frequent and rare words:

  $$f(w) = C / r(w)$$

  or equivalently: $f(w) * r(w) = C$

- Frequency of lexical items (words types) in a large corpus is inversely proportional to their rank.

- Empirical observation in many different corpora

- Brown corpus:
  - half of all types are hapax legomena

Effects of Zipf's Law

- Lexicography:
  - Sinclair (2005): need at least 20 instances
  - BNC (10^8 Tokens): <14% of words appear 20 times or more
- Speech synthesis:
  - may accept bad output for rare words
  - but most words are rare! (at least 1 per sentence)
- Vocabulary growth:
  - vocabulary growth of corpora is not constant
  - G = #hapaxes / #tokens

Back to Language Models

Independence assumptions

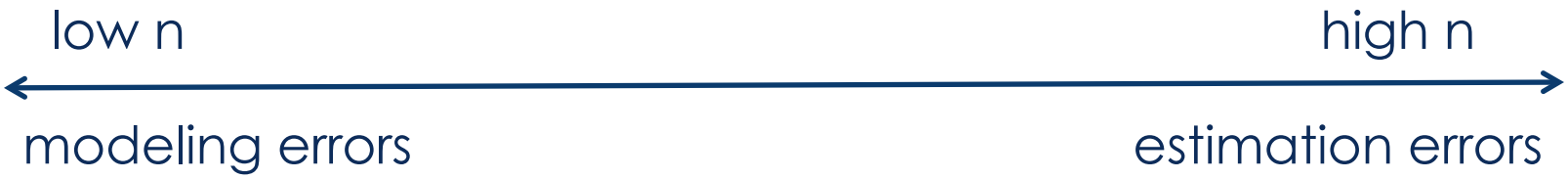
- Let's pretend that word at position t depends only on the words at positions t-1, t-2, ..., t-k for some fixed k (*Markov assumption* of degree k).
- Then we get an n-gram model, with n = k+1:

$$P(X_t \mid X_1,\ldots,X_{t-1}) = P(X_t \mid X_{t-k},\ldots,X_{t-1}) \qquad \text{for all t.}$$

- Special names for *unigram models* (n = 1), *bigram models* (n = 2), *trigram models* (n = 3).

Independence assumption

- We assume independence of X_t from events that are too far in the past, although we know that this assumption is incorrect.
- Typical tradeoff in statistical NLP:
  - if model is too shallow, it won't represent important linguistic dependencies
  - if model is too complex, its parameters can't be estimated accurately from the available data



Tradeoff in practice

In person	*she*		*was*		*inferior*		*to*		*both*		*sisters*	
2-gram	$P(\cdot \mid person)$		$P(\cdot \mid she)$		$P(\cdot \mid was)$		$P(\cdot \mid inferior)$		$P(\cdot \mid to)$		$P(\cdot \mid both)$	
1	and	0.099	had	0.141	not	0.065	**to**	**0.212**	be	0.111	of	0.066
2	who	0.099	**was**	**0.122**	a	0.052			the	0.057	to	0.041
3	to	0.076			the	0.033			her	0.048	in	0.038
4	in	0.045			to	0.031			have	0.027	and	0.025
...												
23	**she**	**0.009**							Mrs	0.006	she	0.009
...												
41									what	0.004	**sisters**	**0.006**
...												
293									**both**	**0.0004**		
...												
∞					**inferior**	**0**						

(Manning/Schütze, ch. 6)

Tradeoff in practice

In person	*she*	*was*		*inferior*		*to*	*both*		*sisters*	
3-gram	$P(\cdot \mid In,person)$	$P(\cdot \mid person,she)$		$P(\cdot \mid she,was)$		$P(\cdot \mid was,inf.)$	$P(\cdot \mid inferior,to)$		$P(\cdot \mid to,both)$	
1	UNSEEN	did	0.5	not	0.057	UNSEEN	the	0.286	to	0.222
2		**was**	**0.5**	very	0.038		Maria	0.143	Chapter	0.111
3				in	0.030		cherries	0.143	Hour	0.111
4				to	0.026		her	0.143	Twice	0.111
...										
∞				**inferior**	**0**		**both**	**0**	**sisters**	**0**

(Manning/Schütze, ch. 6)

Tradeoff in practice

In person	*she*	*was*	*inferior*	*to*	*both*	*sisters*
4-gram	$P(\cdot \mid u,I,p)$	$P(\cdot \mid I,p,s)$	$P(\cdot \mid p,s,w)$	$P(\cdot \mid s,w,i)$	$P(\cdot \mid w,i,t)$	$P(\cdot \mid i,t,b)$
1	UNSEEN	UNSEEN	in 1.0	UNSEEN	UNSEEN	UNSEEN
...						
∞			**inferior 0**			

(Manning/Schütze, ch. 6)

Conclusion

- Statistical models of natural language
- Language models using n-grams
- Data sparseness is a problem.

next Tuesday

- smoothing language models