Language Models

Hongning Wang
CS@UVa

Recap: document generation model

$$Odd(R=1|Q,D) \propto \frac{P(Q,D|R=1)}{P(Q,D|R=0)}$$

$$= \frac{P(D|Q,R=1)}{P(D|Q,R=0)} \boxed{\frac{P(Q|R=1)}{P(Q|R=0)}}$$ ← **Ignored for ranking**

$$\propto \frac{P(D|Q,R=1)}{P(D|Q,R=0)}$$

Model of relevant docs for Q

Model of non-relevant docs for Q

Assume <u>independent</u> attributes of $A_1 \ldots A_k$ ....(why?)

Let $D=d_1 \ldots d_k$, where $d_k \in \{0,1\}$ is the value of attribute A_k (Similarly $Q=q_1 \ldots q_k$)

$$Odd(R=1|Q,D) \propto \prod_{i=1}^{k} \frac{P(A_i=d_i|Q,R=1)}{P(A_i=d_i|Q,R=0)}$$

Terms occur in doc

Terms do not occur in doc

$$= \prod_{i=1,d_i=1}^{k} \frac{P(A_i=1|Q,R=1)}{P(A_i=1|Q,R=0)} \prod_{i=1,d_i=0}^{k} \frac{P(A_i=0|Q,R=1)}{P(A_i=0|Q,R=0)}$$

document	relevant(R=1)	nonrelevant(R=0)
term present A_i=1	p_i	u_i
term absent A_i=0	$1-p_i$	$1-u_i$

Recap: document generation model

$$Odd(R=1|Q,D) \propto \prod_{i=1}^{k} \frac{P(A_i = d_i | Q, R=1)}{P(A_i = d_i | Q, R=0)}$$

Terms occur in doc

Terms do not occur in doc

$$= \boxed{\prod_{i=1, d_i=1}^{k} \frac{P(A_i = 1 | Q, R=1)}{P(A_i = 1 | Q, R=0)}} \boxed{\prod_{i=1, d_i=0}^{k} \frac{P(A_i = 0 | Q, R=1)}{P(A_i = 0 | Q, R=0)}}$$

Important tricks

$$\approx \prod_{i=1, d_i=q_i=1}^{k} \frac{p_i}{u_i} \prod_{i=1, d_i=0, q_i=1}^{k} \frac{1-p_i}{1-u_i}$$

Assumption: terms not occurring in the query are equally likely to occur in relevant and nonrelevant documents, i.e., $p_t = u_t$

$$= \prod_{i=1, d_i=q_i=1}^{k} \frac{p_i(1-u_i)}{u_i(1-p_i)} \cancel{\prod_{i=1, q_i=1}^{k} \frac{1-p_i}{1-u_i}}$$

document	relevant(R=1)	nonrelevant(R=0)
term present A_i=1	p_i	u_i
term absent A_i=0	1-p_i	1-u_i

Recap: Maximum likelihood vs. Bayesian

- Maximum likelihood estimation
  - "Best" means "data likelihood reaches maximum"

$$\widehat{\boldsymbol{\theta}} = \mathbf{argmax}_{\boldsymbol{\theta}} \mathbf{P}(\mathbf{X}|\boldsymbol{\theta})$$

  - Issue: small sample size

ML: Frequentist's point of view

- Bayesian estimation
  - "Best" means being consistent with our "prior" knowledge and explaining data well

$$\widehat{\boldsymbol{\theta}} = \mathbf{argmax}_{\boldsymbol{\theta}} \boldsymbol{P}(\boldsymbol{\theta}|\boldsymbol{X}) = \mathbf{argmax}_{\boldsymbol{\theta}} \mathbf{P}(\mathbf{X}|\boldsymbol{\theta})\mathbf{P}(\boldsymbol{\theta})$$

  - A.k.a, Maximum a Posterior estimation
  - Issue: how to define prior?

MAP: Bayesian's point of view

Recap: Robertson-Sparck Jones Model

(Robertson & Sparck Jones 76)

$$\log O(R=1|Q,D) \stackrel{Rank}{\approx} \sum_{i=1,d_i=q_i=1}^{k} \log \frac{p_i(1-u_i)}{u_i(1-p_i)} = \sum_{i=1,d_i=q_i=1}^{k} \log \frac{p_i}{1-p_i} + \log \frac{1-u_i}{u_i}$$ (RSJ model)

Two parameters for each term A_i:

p_i = P(A_i=1|Q,R=1): prob. that term A_i occurs in a relevant doc

u_i = P(A_i=1|Q,R=0): prob. that term A_i occurs in a non-relevant doc

How to estimate these parameters?

Suppose we have relevance judgments,

$$\hat{p}_i = \frac{\#(rel.\ doc\ with\ A_i) + 0.5}{\#(rel.doc) + 1} \qquad \hat{u}_i = \frac{\#(nonrel.\ doc\ with\ A_i) + 0.5}{\#(nonrel.doc) + 1}$$

- "+0.5" and "+1" can be justified by Bayesian estimation as priors

Per-query estimation!

Recap: the BM25 formula

- A closer look

TF-IDF component for document

TF component for query

$$rel(q,D)=\sum_{i=1}^{n}\boxed{IDF(q_i)\frac{tf_i(k_1+1)}{tf_i+k_1(1-b+b\frac{|D|}{avg|D|})}}\boxed{\frac{qtf_i(k_2+1)}{k_2+qtf_i}}$$

  - b is usually set to [0.75, 1.2]
  - k_1 is usually set to [1.2, 2.0]
  - k_2 is usually set to (0, 1000]

Vector space model with TF-IDF schema!

Notion of Relevance

- Relevance
  - Δ(Rep(q), Rep(d)) **Similarity**
    - **Different rep & similarity**
      - **Vector space model** (Salton et al., 75)
      - …
      - **Prob. distr. model** (Wong & Yao, 89)
  - **P(r=1|q,d) $r \in \{0,1\}$ Probability of Relevance**
    - **Regression Model** (Fox 83)
    - **Generative Model**
      - **Doc generation**
        - **Classical prob. Model** (Robertson & Sparck Jones, 76)
      - **Query generation**
        - **LM approach** (Ponte & Croft, 98) (Lafferty & Zhai, 01a)
  - **P(d $\rightarrow$q) or P(q $\rightarrow$d) Probabilistic inference**
    - **Different inference system**
      - **Prob. concept space model** (Wong & Yao, 95)
      - **Inference network model** (Turtle & Croft, 91)

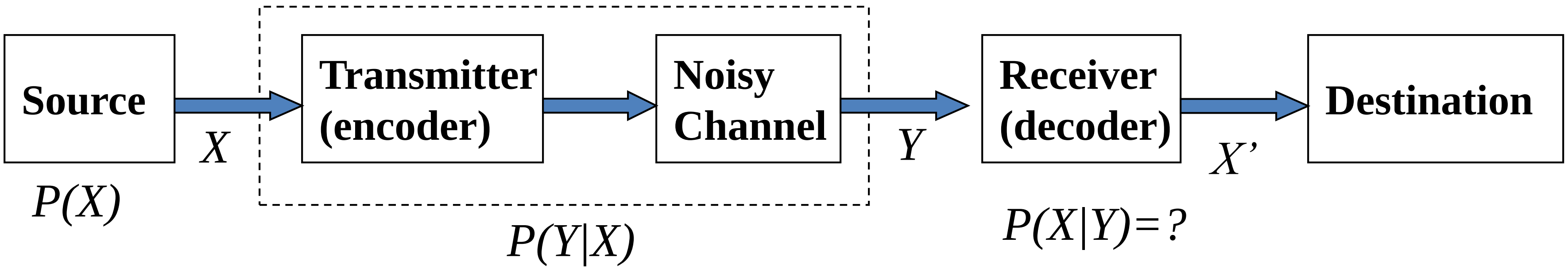
What is a statistical LM?

- A model specifying probability distribution over word sequences
  - p("*Today is Wednesday*") ≈ 0.001
  - p("*Today Wednesday is*") ≈ 0.0000000000001
  - p("*The eigenvalue is positive*") ≈ 0.00001
- It can be regarded as a probabilistic mechanism for "generating" text, thus also called a "generative" model

Why is a LM useful?

- Provides a principled way to quantify the uncertainties associated with natural language
- Allows us to answer questions like:
  – Given that we see "*John*" and "*feels*", how likely will we see "*happy*" as opposed to "*habit*" as the next word?
    (speech recognition)
  – Given that we observe "baseball" three times and "game" once in a news article, how likely is it about "sports"?
    (text categorization, information retrieval)
  – Given that a user is interested in sports news, how likely would the user use "baseball" in a query?
    (information retrieval)

Source-Channel framework [Shannon 48]



$$\hat{X} = \arg\max_X \; p(X \mid Y) = \arg\max_X \; p(Y \mid X)\, p(X) \quad \textbf{(Bayes Rule)}$$

When X is text, p(X) is a language model

Many Examples:

Speech recognition:	**X=Word sequence**	**Y=Speech signal**
Machine translation:	**X=English sentence**	**Y=Chinese sentence**
OCR Error Correction:	**X=Correct word**	**Y= Erroneous word**
Information Retrieval:	**X=Document**	**Y=Query**
Summarization:	**X=Summary**	**Y=Document**

Language model for text

We need independence assumptions!

- Probability distribution over word sequences
  - $p(w_1\, w_2\ \dots\ w_n) =$
    $p(w_1)p(w_2|w_1)p(w_3|w_1,w_2)\dots p(w_n|w_1,w_2,\dots,w_{n-1})$
  - Complexity - $O(V^{n^*})$
    - n^* - maximum ~~document~~ length
      sentence

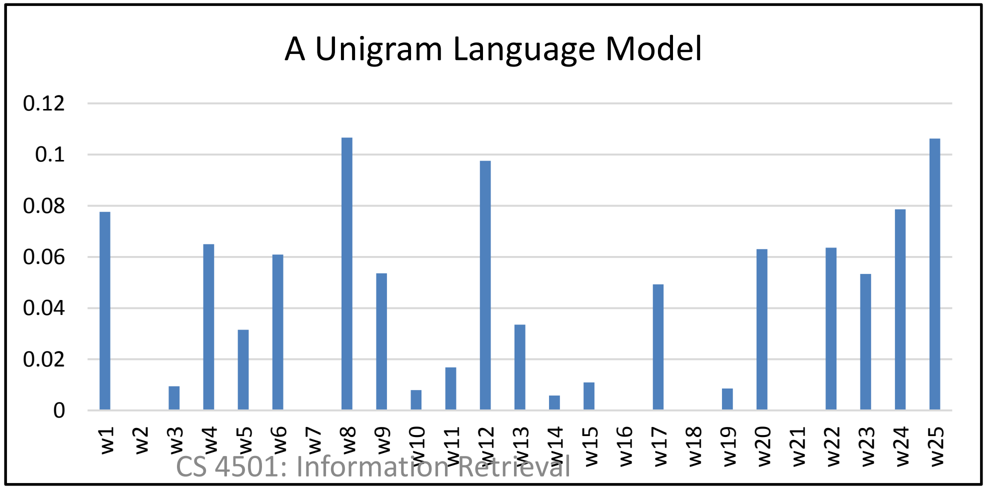
Chain rule: from conditional probability to joint probability

- 475,000 main headwords in Webster's Third New International Dictionary
- Average English sentence length is 14.3 words
- A rough estimate: $O(475000^{14})$

How large is this? $\frac{475000^{14}}{8bytes \times (1024)^4} \approx 3.38e^{66}TB$

Unigram language model

- Generate a piece of text by generating each word <u>independently</u>
  - $p(w_1\, w_2\ \dots\ w_n) = p(w_1)p(w_2)\dots p(w_n)$
  - $s.t. \{p(w_i)\}_{i=1}^{N}, \sum_i p(w_i) = 1, p(w_i) \geq 0$
- Essentially a multinomial distribution over the vocabulary



The simplest and most popular choice!

More sophisticated LMs

- N-gram language models
  - In general, $p(w_1\, w_2\ \dots w_n) = p(w_1)p(w_2|w_1) \dots p(w_n|w_1 \dots w_{n-1})$
  - N-gram: conditioned only on the past N-1 words
  - E.g., bigram: $p(w_1\ \dots\ w_n) = p(w_1)p(w_2|w_1)\, p(w_3|w_2)\ \dots p(w_n|w_{n-1})$
- Remote-dependence language models (e.g., Maximum Entropy model)
- Structured language models (e.g., probabilistic context-free grammar)

Why just unigram models?

- Difficulty in moving toward more complex models
  - They involve more parameters, so need more data to estimate
  - They increase the computational complexity significantly, both in time and space
- Capturing word order or structure may not add so much value for “topical inference”
- But, using more sophisticated models can still be expected to improve performance ...

Generative view of text documents

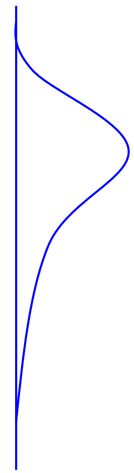
(Unigram) Language Model θ
$p(w|\ \theta)$

Sampling

Document

Topic 1:
Text mining

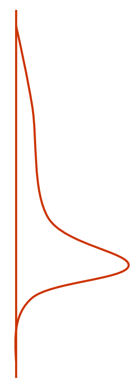
…
text 0.2
mining 0.1
assocation 0.01
clustering 0.02
…
food 0.00001
…



Text mining document

Topic 2:
Health

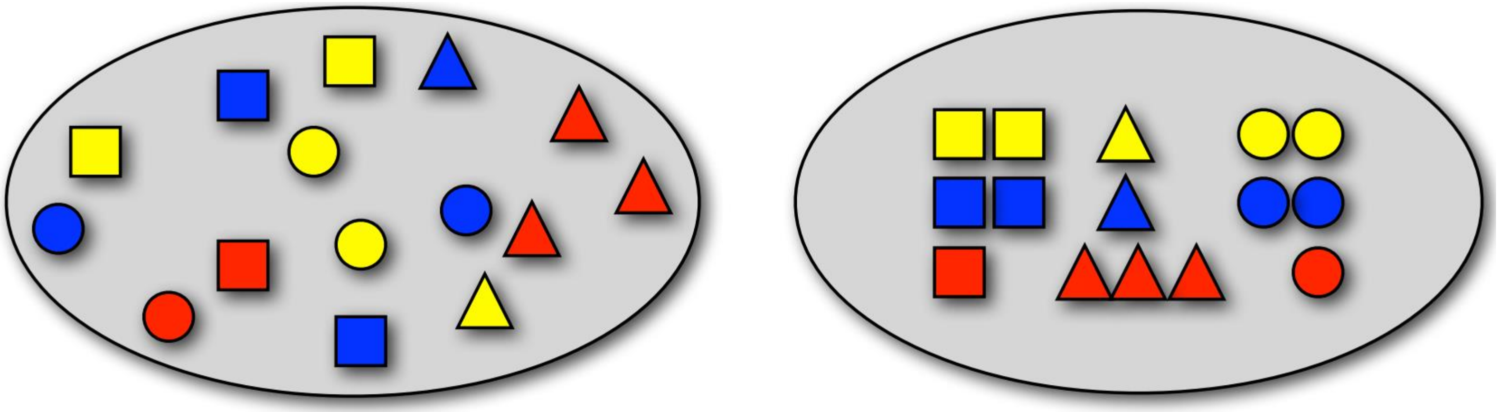
…
text 0.01
food 0.25
nutrition 0.1
healthy 0.05
diet 0.02
…



Food nutrition document

Sampling with replacement

- Pick a random shape, then put it back in the bag



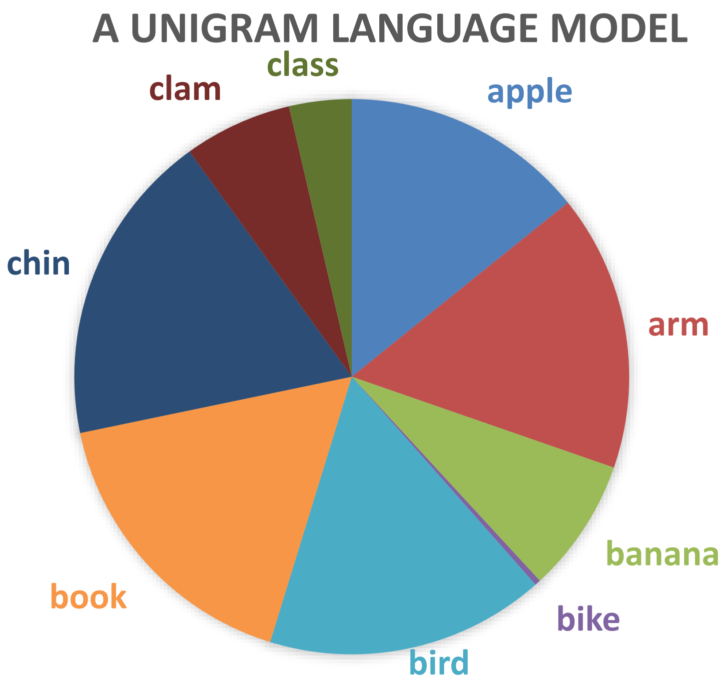
P(■) = 2/15
P(blue) = 5/15
P(blue | □) = 2/5

P(■) = 1/15
P(red) = 5/15
P(□) = 5/15

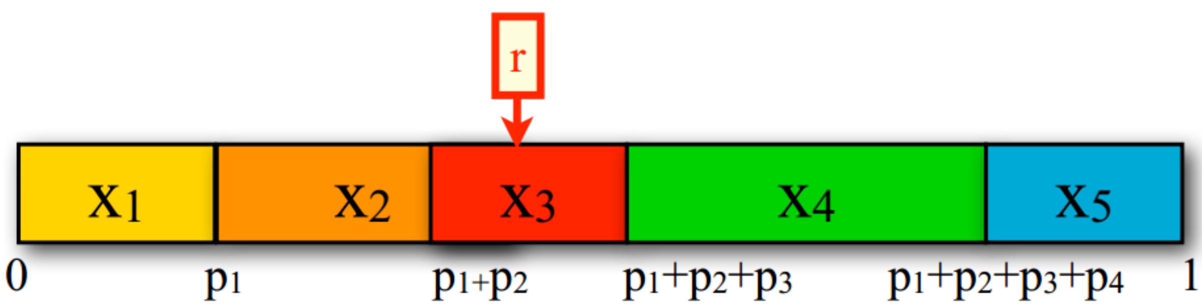
P(■ or ▲) = 2/15
P(△ | red) = 3/5

How to generate text document from an N-gram language model?

- Sample from a discrete distribution $p(X)$



s into



Generating text from language models

P(of) = 3/66
P(Alice) = 2/66
P(was) = 2/66
P(to) = 2/66
P(her) = 2/66
P(sister) = 2/66
P(,) = 4/66
P(') = 4/66

Under a unigram language model:

Alice was beginning to get very tired of sitting by her sister on the bank, and of having nothing to do: once or twice she had peeped into the book her sister was reading, but it had no pictures or conversations in it, 'and what is the use of a book,' thought Alice 'without pictures or conversation?'

Generating text from language models

P(of) = 3/66
P(Alice) = 2/66
P(was) = 2/66
P(to) = 2/66
P(her) = 2/66
P(sister) = 2/66
P(,) = 4/66
P(') = 4/66

Under a unigram language model:

The same likelihood!

beginning by, very Alice but was and?
reading no tired of to into sitting
sister the, bank, and thought of without
her nothing: having conversations Alice
once do or on she it get the book her had
peeped was conversation it pictures or
sister in, 'what is the use had twice of
a book''pictures or' to

N-gram language models will help

Generated from language models of New York Times

- Unigram
  - Months the my and issue of year foreign new exchange's september were recession exchange new endorsed a q acquire to six executives.
- Bigram
  - Last December through the way to preserve the Hudson corporation N.B.E.C. Taylor would seem to complete the major central planners one point five percent of U.S.E. has already told M.X. corporation of living on information such as more frequently fishing to keep her.
- Trigram
  - They also point to ninety nine point six billon dollars from two hundred four oh six three percent of the rates of interest stores as Mexico and Brazil on market conditions.

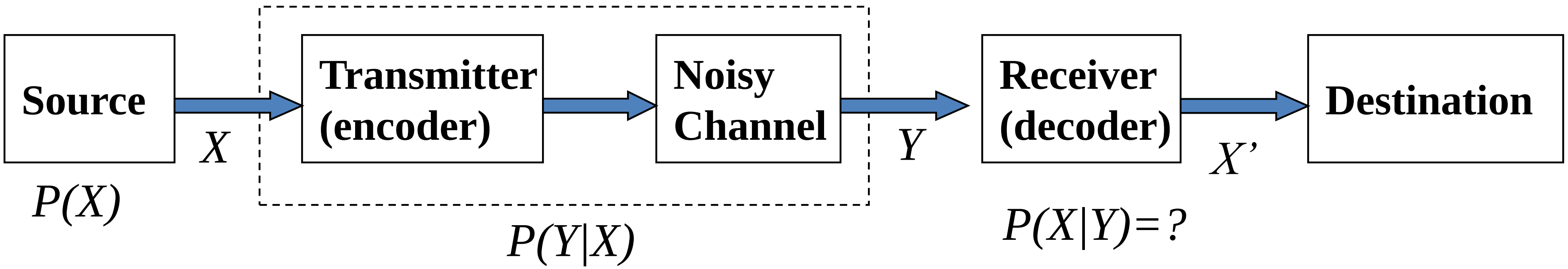
Turing test: generating Shakespeare

A	• To him swallowed confess hear both. Which. Of save on trail for are ay device and rote life have • Every enter now severally so, let • Hill he late speaks; or! a more to leg less first you enter • Are where exeunt and sighs have rise excellency took of.. Sleep knave we. near; vile like
B	• What means, sir. I confess she? then all sorts, he is trim, captain. •Why dost stand forth thy canopy, forsooth; he is this palpable hit the King Henry. Live king. Follow. •What we, hath got so she that I rest and sent to scold and nature bankrupt, nor the firs[illegible] •E[illegible] [SCIgen - An Automatic CS Paper Generator] [illegible]com-mand of fear not a liberal largess given away, Falstaff! Exeunt
C	• Sweet prince, Falstaff shall die. Harry of Monmouth's grave. • This shall forbid it should be branded, if renown made it empty. • Indeed the duke; and had a very good friend. • Fly, and will rid me these news of price. Therefore the sadness of parting, as they say, 'tis done.
D	• King Henry. What! I will go seek the traitor Gloucester. Exeunt some of the watch. A great banquet serv'd in; • Will you not tell me who I am? • It cannot be but so. • Indeed the short and the long. Marry, 'tis a noble Lepidus.

Recap: what is a statistical LM?

- A model specifying probability distribution over word sequences
  - p("*Today is Wednesday*") ≈ 0.001
  - p("*Today Wednesday is*") ≈ 0.0000000000001
  - p("*The eigenvalue is positive*") ≈ 0.00001
- It can be regarded as a probabilistic mechanism for "generating" text, thus also called a "generative" model

Recap: Source-Channel framework
[Shannon 48]



$$\hat{X} = \arg\max_{X} \; p(X \mid Y) = \arg\max_{X} \; p(Y \mid X)\, p(X)$$ **(Bayes Rule)**

When X is text, p(X) is a language model

Many Examples:

Speech recognition:	**X=Word sequence**	**Y=Speech signal**
Machine translation:	**X=English sentence**	**Y=Chinese sentence**
OCR Error Correction:	**X=Correct word**	**Y= Erroneous word**
Information Retrieval:	**X=Document**	**Y=Query**
Summarization:	**X=Summary**	**Y=Document**

Recap: language model for text

We need independence assumptions!

- Probability distribution over <u>word sequences</u>
  - $p(w_1\ w_2\ \ldots\ w_n) =$
    $p(w_1)p(w_2|w_1)p(w_3|w_1,w_2)\ldots p(w_n|w_1,w_2,\ldots,w_{n-1})$
  - Complexity - $O(V^{n^*})$
    - n^* - maximum ~~document~~ length
      sentence

Chain rule: from conditional probability to joint probability

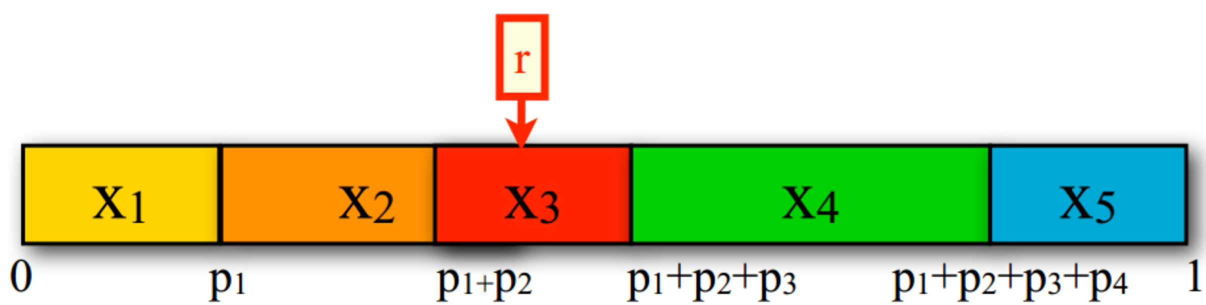
- 475,000 main headwords in Webster's Third New International Dictionary
- Average English sentence length is 14.3 words
- A rough estimate: $O(475000^{14})$

How large is this? $\frac{475000^{14}}{8bytes \times (1024)^4} \approx 3.38e^{66}TB$

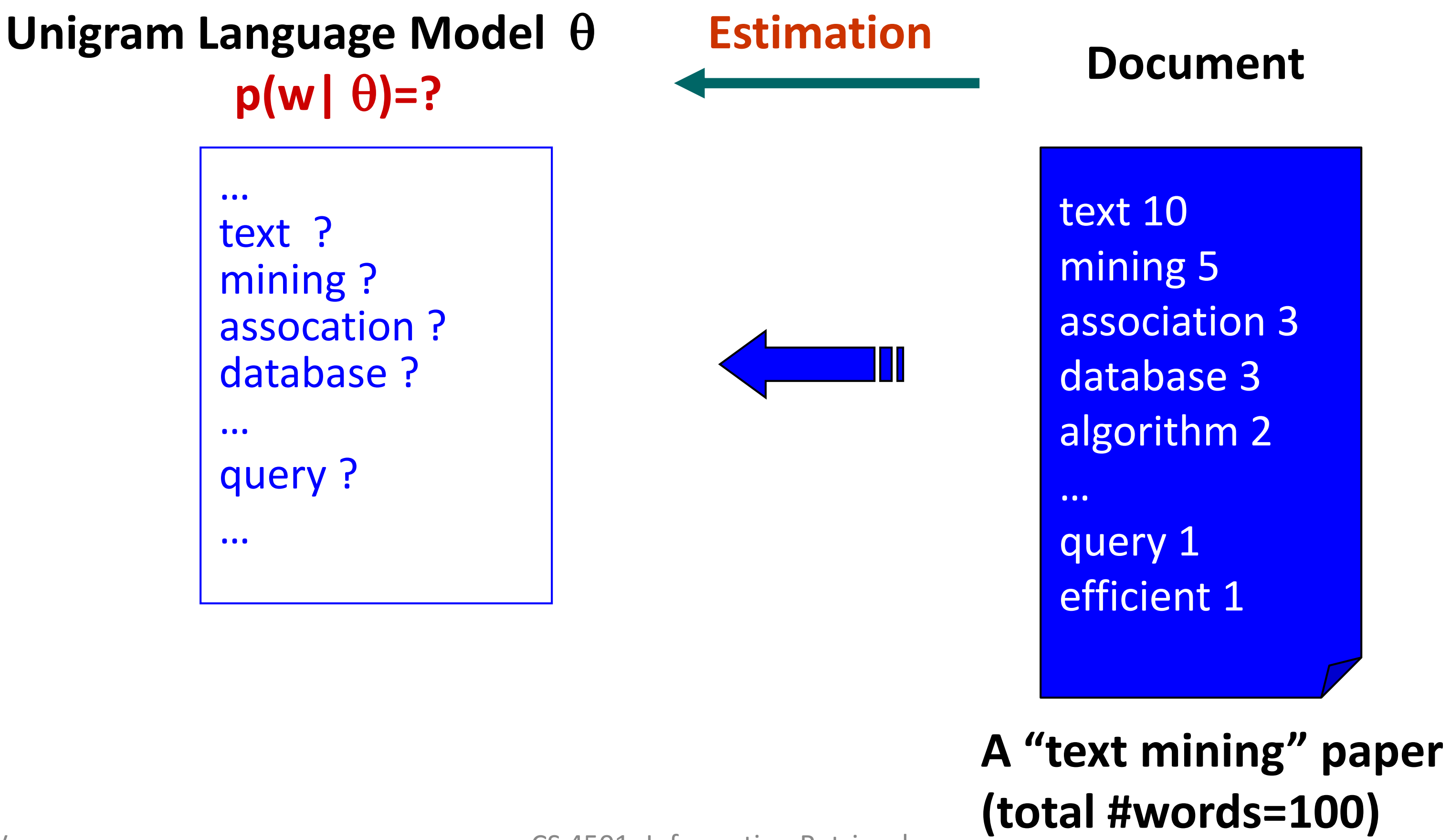
Recap: how to generate text document from an N-gram language model?

- Sample from a <u>discrete</u> distribution $p(X)$
  - Assume n outcomes in the event space X

1. Divide the interval [0,1] into n intervals according to the probabilities of the outcomes
2. Generate a random number r **uniformly** between 0 and 1
3. Return x_i where r falls into

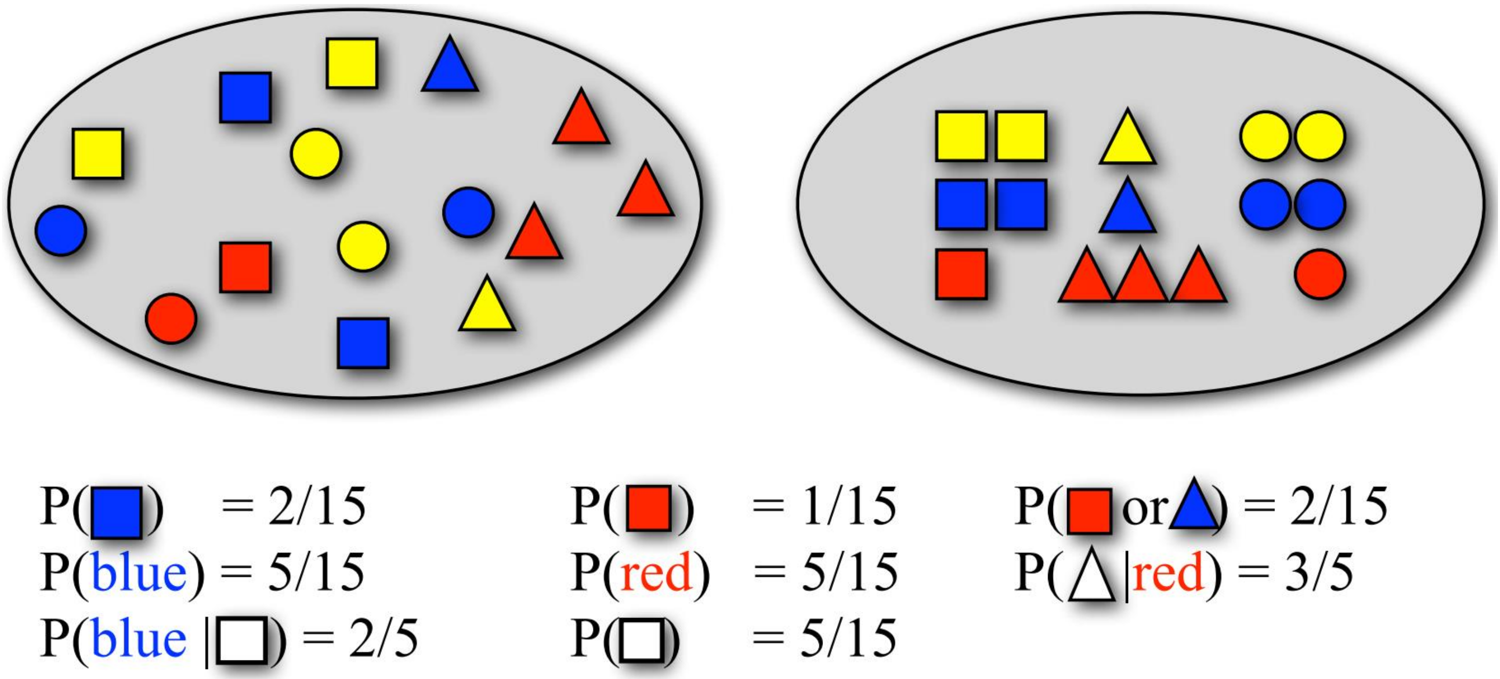


Estimation of language models



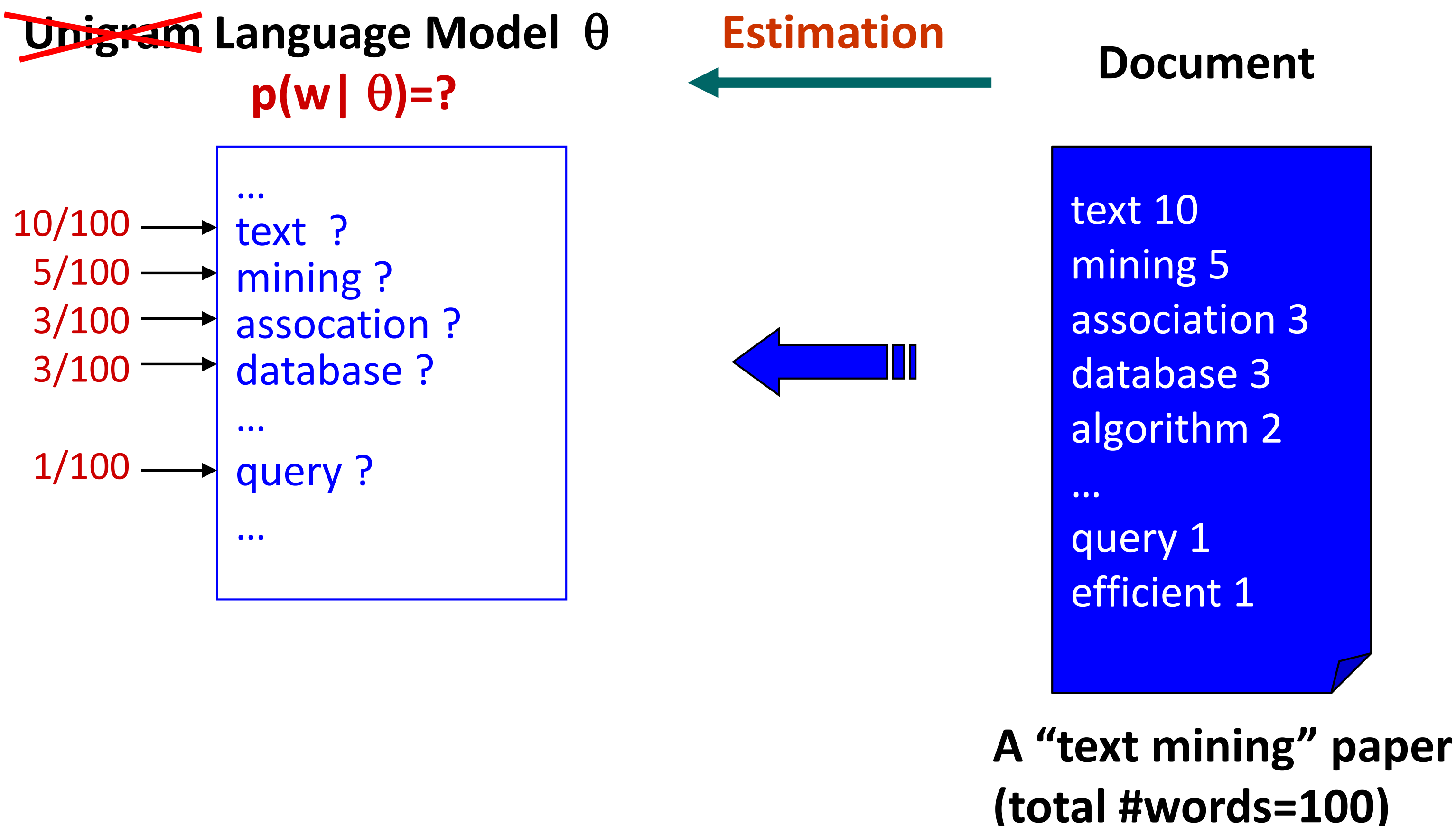
Sampling with replacement

- Pick a random shape, then put it back in the bag



Estimation of language models

- Maximum likelihood estimation



Maximum likelihood estimation

- For N-gram language models
  - $p(w_i|w_{i-n+1}, \dots, w_{i-1}) = \frac{c(w_{i-n+1}, \dots, w_{i-1}, w_i)}{c(w_{i-n+1}, \dots, w_{i-1})}$
    - $c(\emptyset) = N$

Length of document or total number of words in a corpus

Language models for IR
[Ponte & Croft SIGIR'98]

Document

Language Model

Query

Text mining paper

…
text ?
mining ?
assocation ?
clustering ?
…
food ?
…

?

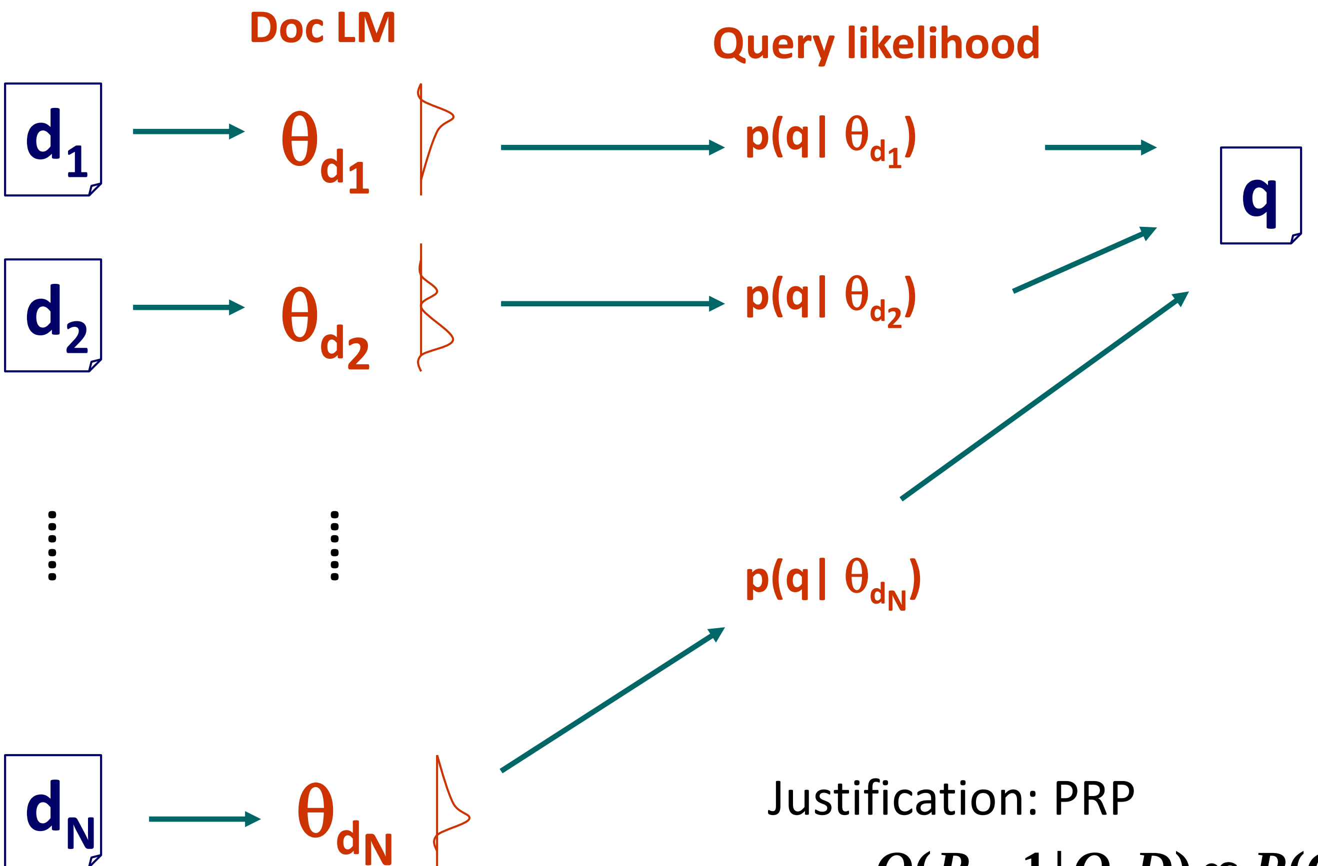
"data mining algorithms"

Food nutrition paper

…
food ?
nutrition ?
healthy ?
diet ?
…

Which model would most likely have generated this query?

Ranking docs by query likelihood



Justification: PRP

$$O(R=1|Q,D) \propto P(Q|D,R=1)$$

Justification from PRP

$$O(R=1|Q,D) \propto \frac{P(Q,D|R=1)}{P(Q,D|R=0)}$$

Query generation

$$= \frac{P(Q|D,R=1)P(D|R=1)}{P(Q|D,R=0)P(D|R=0)}$$

$$\propto \underline{P(Q|D,R=1)}\underline{\frac{P(D|R=1)}{P(D|R=0)}} \quad (Assume\ P(Q|D,R=0) \approx P(Q|R=0))$$

Query likelihood p(q| θ_d) **Document prior**

Assuming uniform document prior, we have

$$O(R=1|Q,D) \propto P(Q|D,R=1)$$

Retrieval as language model estimation

- Document ranking based on *query likelihood*

$$\log p(q \mid d) = \sum_i \log \boxed{p(w_i \mid d)}$$

Document language model

$$where,\ \ q = w_1 w_2 \ldots w_n$$

  - Retrieval problem $\approx$ Estimation of $p(w_i|d)$
  - Common approach
    - Maximum likelihood estimation (MLE)

Problem with MLE

- Unseen events

  – There a ... ction of Yelp re...

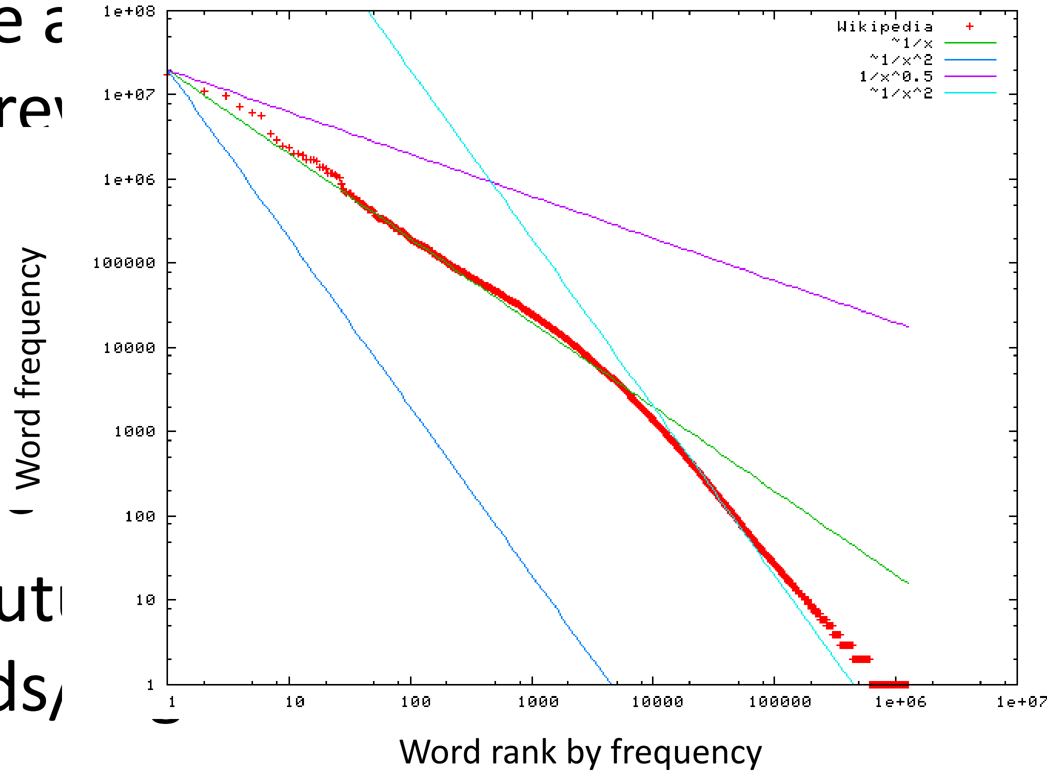
    - Or

    - Or ... d

  ➢ This ... not occur in th ... th MLE!

  ➢ No fut ... e unseen words, ...

$f(k; s, N) \propto 1/k^s$



A plot of word frequency in Wikipedia (Nov 27, 2006)

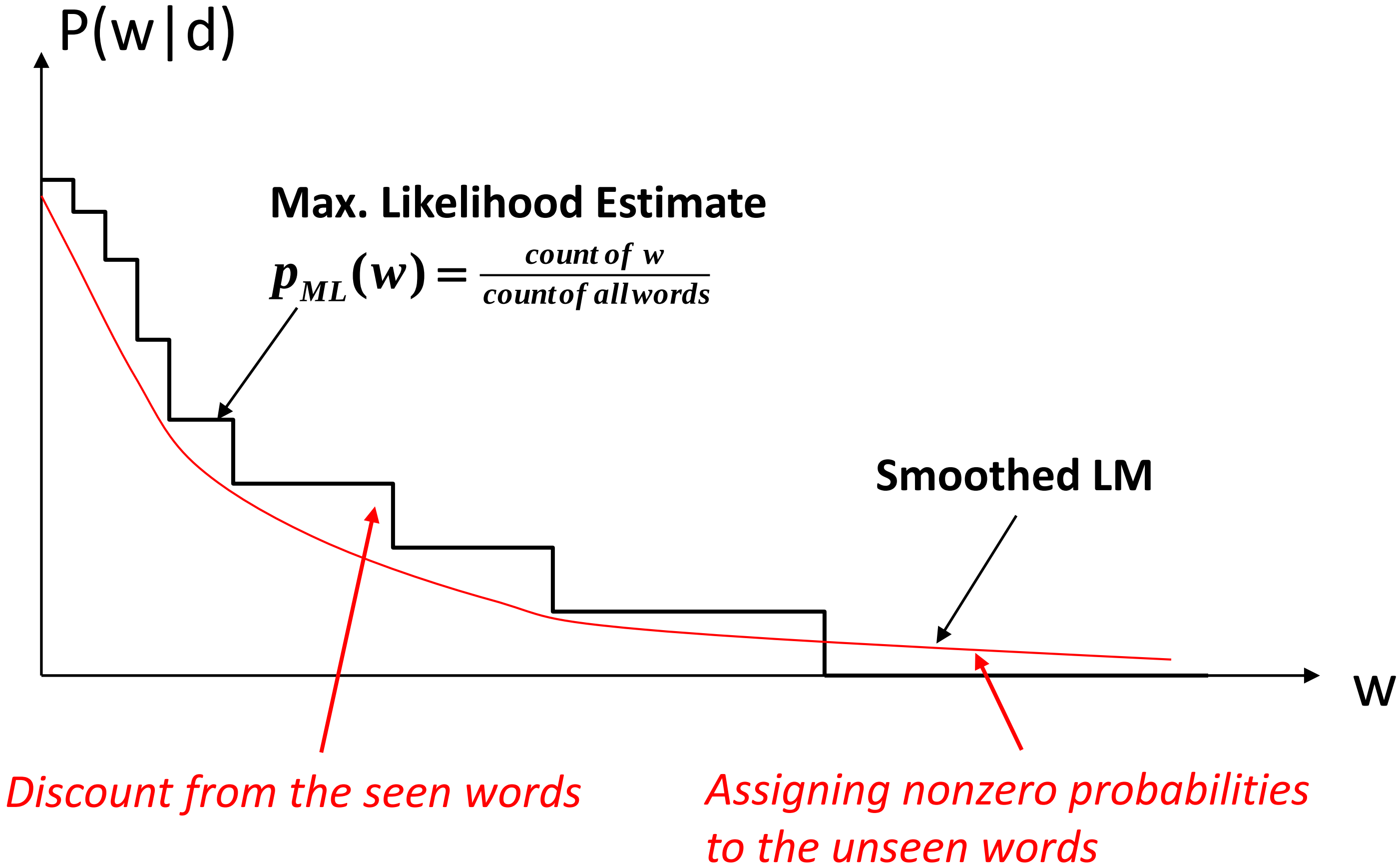
Problem with MLE

- What probability should we give a word that has not been observed in the document?
  - log0?
- If we want to assign non-zero probabilities to such words, we'll have to discount the probabilities of observed words
- This is so-called "smoothing"

General idea of smoothing

- All smoothing methods try to
  1. Discount the probability of words seen in a document
  2. Re-allocate the extra counts such that unseen words will have a non-zero count

Illustration of language model smoothing



Smoothing methods

- Method 1: Additive smoothing
  - Add a constant δ to the counts of each word

Counts of w in d

$$p(w|d) = \frac{c(w,d)+1}{|d|+|V|}$$

"Add one", Laplace smoothing

Vocabulary size

Length of d (total counts)

  - Problems?
    - Hint: all words are equally important?

Add one smoothing for bigrams

Original:

	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

Smoothed:

	i	want	to	eat	chinese	food	lunch	spend
i	6	828	1	10	1	1	1	3
want	3	1	609	2	7	7	6	2
to	3	1	5	687	3	1	7	212
eat	1	1	3	1	17	3	43	1
chinese	2	1	1	1	1	83	2	1
food	16	1	16	1	2	5	1	1
lunch	3	1	1	1	1	2	1	1
spend	2	1	2	1	1	1	1	1

After smoothing

- Giving too much to the unseen events

Original:

	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

Smoothed:

	i	want	to	eat	chinese	food	lunch	spend
i	0.0015	0.21	0.00025	0.0025	0.00025	0.00025	0.00025	0.00075
want	0.0013	0.00042	0.26	0.00084	0.0029	0.0029	0.0025	0.00084
to	0.00078	0.00026	0.0013	0.18	0.00078	0.00026	0.0018	0.055
eat	0.00046	0.00046	0.0014	0.00046	0.0078	0.0014	0.02	0.00046
chinese	0.0012	0.00062	0.00062	0.00062	0.00062	0.052	0.0012	0.00062
food	0.0063	0.00039	0.0063	0.00039	0.00079	0.002	0.00039	0.00039
lunch	0.0017	0.00056	0.00056	0.00056	0.00056	0.0011	0.00056	0.00056
spend	0.0012	0.00058	0.0012	0.00058	0.00058	0.00058	0.00058	0.00058

Refine the idea of smoothing

- Should all unseen words get equal probabilities?
- We can use a reference model to discriminate unseen words

Discounted ML estimate

$$p(w|d)=\begin{cases} p_{seen}(w|d) & \text{if } w \text{ is seen in } d \\ \alpha_d \, p(w|REF) & \text{otherwise} \end{cases}$$

Reference language model

$$\alpha_d = \frac{1-\sum_{w \text{ is seen}} p_{seen}(w|d)}{\sum_{w \text{ is unseen}} p(w|REF)}$$

Smoothing methods

- Method 2: Absolute discounting
  - Subtract a constant δ from the counts of each word

uniq words

$$p(w|d) = \frac{\max(c(w;d)-\delta,0)+\delta|d|_u\, p(w|REF)}{|d|}$$

  - Problems?
    - Hint: varied document length?

Smoothing methods

- Method 3: Linear interpolation, Jelinek-Mercer
  - "Shrink" uniformly toward p(w|REF)

$$p(w \mid d) = (1-\lambda)\underbrace{\boxed{\frac{c(w,d)}{|d|}}}_{\text{MLE}} + \underbrace{\lambda}_{\text{parameter}} p(w \mid REF)$$

  - Problems?
    - Hint: what is missing?

Smoothing methods

- Method 4: Dirichlet Prior/Bayesian
  - Assume pseudo counts $\mu p(w|REF)$

$$p(w|d) = \frac{c(w;d) + \mu p(w|REF)}{|d| + \mu} = \frac{|d|}{|d|+\mu}\frac{c(w,d)}{|d|} + \frac{\mu}{|d|+\mu} p(w|REF)$$

parameter

  - Problems?

Dirichlet prior smoothing

likelihood of doc given the model

prior over models

- Bayesian estimator
  - Posterior of LM: $p(\theta|d) \propto p(d|\theta)p(\theta)$
- Conjugate prior
  - Posterior will be in the same form as prior
  - Prior can be interpreted as “extra”/“pseudo” data
- Dirichlet distribution is a conjugate prior for multinomial distribution

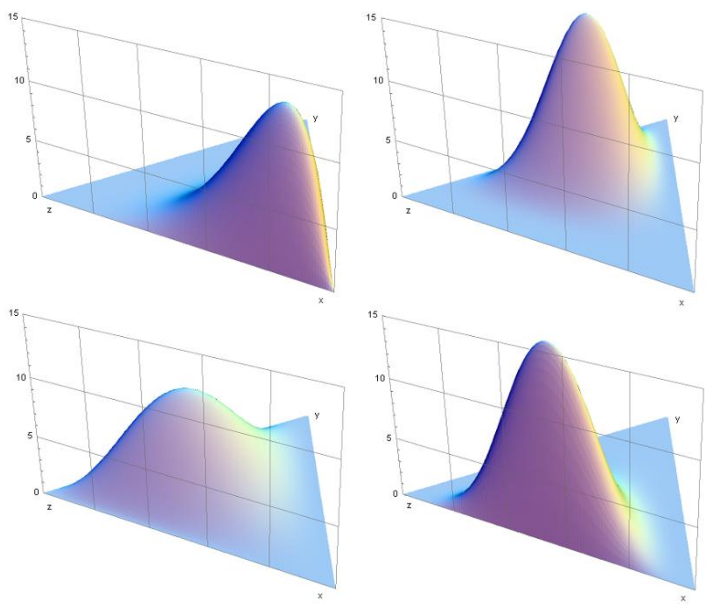
$$Dir(\theta \mid \underline{\alpha_1,\ldots,\alpha_N}) = \frac{\Gamma(\alpha_1+\cdots+\alpha_N)}{\Gamma(\alpha_1)\cdots\Gamma(\alpha_N)}\prod_{i=1}^{N}\theta_i^{\alpha_i-1}$$

“extra”/“pseudo” word counts, we set $\alpha_i=\mu\, p(w_i|REF)$

Some background knowledge

- Conjugate prior
  - Posterior dist in the same family as prior
- Dirichlet distribution
  - Continuous
  - Samples from it will be the parameters in a multinomial distribution

Gaussian -> Gaussian
Beta -> Binomial
Dirichlet -> Multinomial



Dirichlet prior smoothing (cont)

Posterior distribution of parameters:

$$p(\theta|d) = Dir(\theta|c(w_1)+\alpha_1,\dots,c(w_N)+\alpha_N)$$

$$\textbf{Property: If } \theta \sim Dir(\theta|\alpha), \textbf{ then } \mathrm{E}(\theta) = \{\frac{\alpha_i}{\sum \alpha_i}\}$$

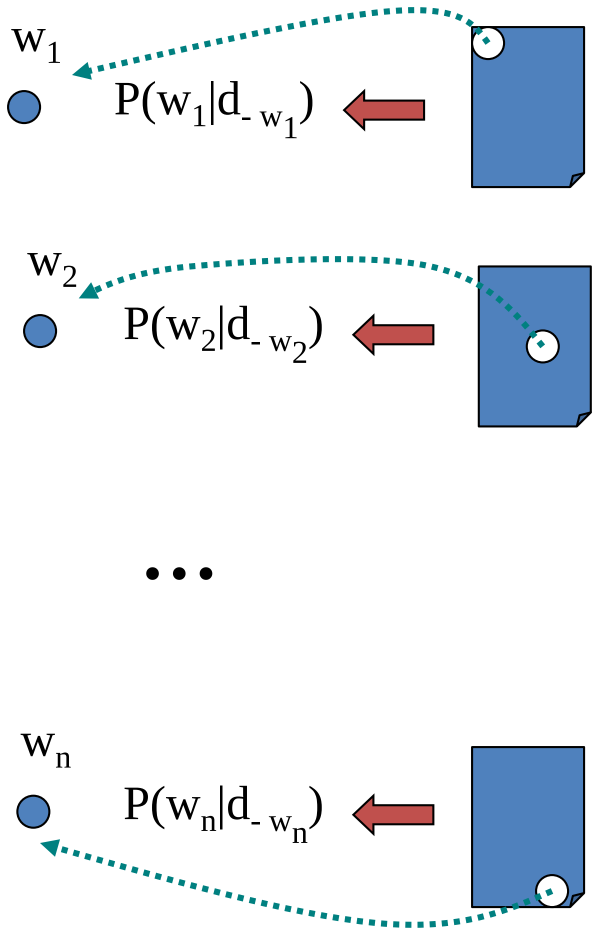
The predictive distribution is the same as the mean:

$$\mathbf{p}(\mathbf{w}_i|\hat{\theta}) = \int \mathbf{p}(\mathbf{w}_i|\theta) Dir(\theta|\alpha) d\theta$$

$$= \frac{c(\mathbf{w}_i)+\alpha_i}{|\mathbf{d}|+\sum_{i=1}^{N}\alpha_i} = \boxed{\frac{c(\mathbf{w}_i)+\mu p(w_i|REF)}{|\mathbf{d}|+\mu}}$$

Dirichlet prior smoothing

Estimating μ using leave-one-out
[Zhai & Lafferty 02]



Leave-one-out

log-likelihood

$$L_{-1}(\mu \mid C) = \sum_{i=1}^{N} \sum_{w \in V} c(w, d_i) \log\left(\frac{c(w, d_i) - 1 + \mu p(w \mid C)}{|d_i| - 1 + \mu}\right)$$

Maximum Likelihood Estimator

$$\hat{\mu} = \underset{\mu}{\operatorname{argmax}} \; L_{-1}(\mu \mid C)$$

Why would “leave-one-out” work?

20 word by author1

abc abc ab c d d
abc cd d d
abd ab ab ab ab
cd d e cd e

Now, suppose we leave “e” out…

μ doesn’t have to be big

$$p_{ml}("e" | author1) = \frac{1}{19} \qquad p_{smooth}("e" | author1) = \frac{20}{20+\mu}\frac{1}{19} + \frac{\mu}{20+\mu} p("e" | REF)$$

20 word by author2

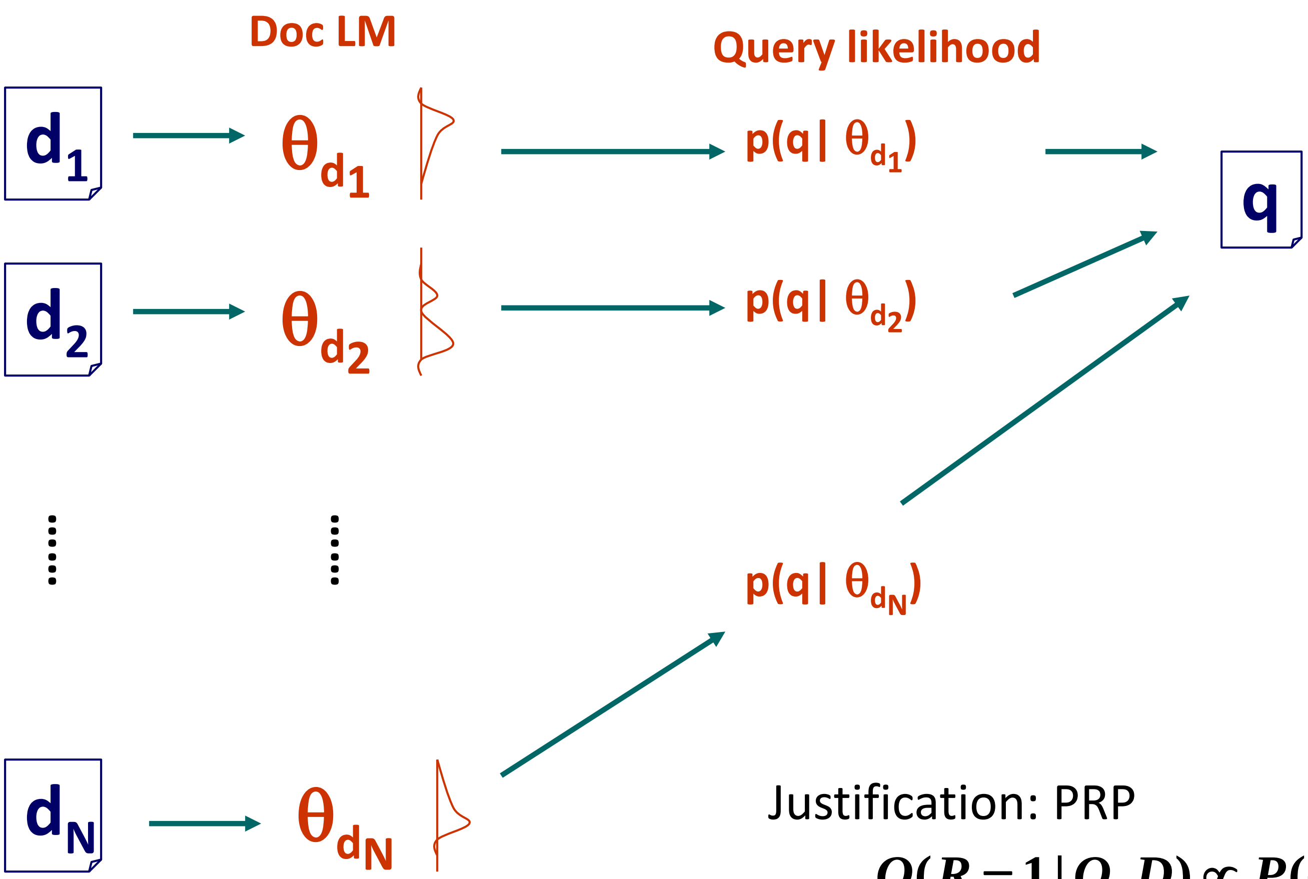
abc abc ab c d d
abe cb e f
acf fb ef aff abef
cdc db ge f s

$$p_{ml}("e" | author2) = \frac{0}{19} \qquad p_{smooth}("e" | author2) = \frac{20}{20+\mu}\frac{0}{19} + \frac{\mu}{20+\mu} p("e" | REF)$$

μ must be big! <u>more smoothing</u>

The amount of smoothing is closely related to the underlying vocabulary size

Recap: ranking docs by query likelihood



Justification: PRP

$$O(R=1|Q,D) \propto P(Q|D,R=1)$$

Recap: retrieval as language model estimation

- Document ranking based on *query likelihood*

$$\log p(q \mid d) = \sum_i \log \boxed{p(w_i \mid d)}$$

$$where,\ \ q = w_1 w_2 \ldots w_n$$

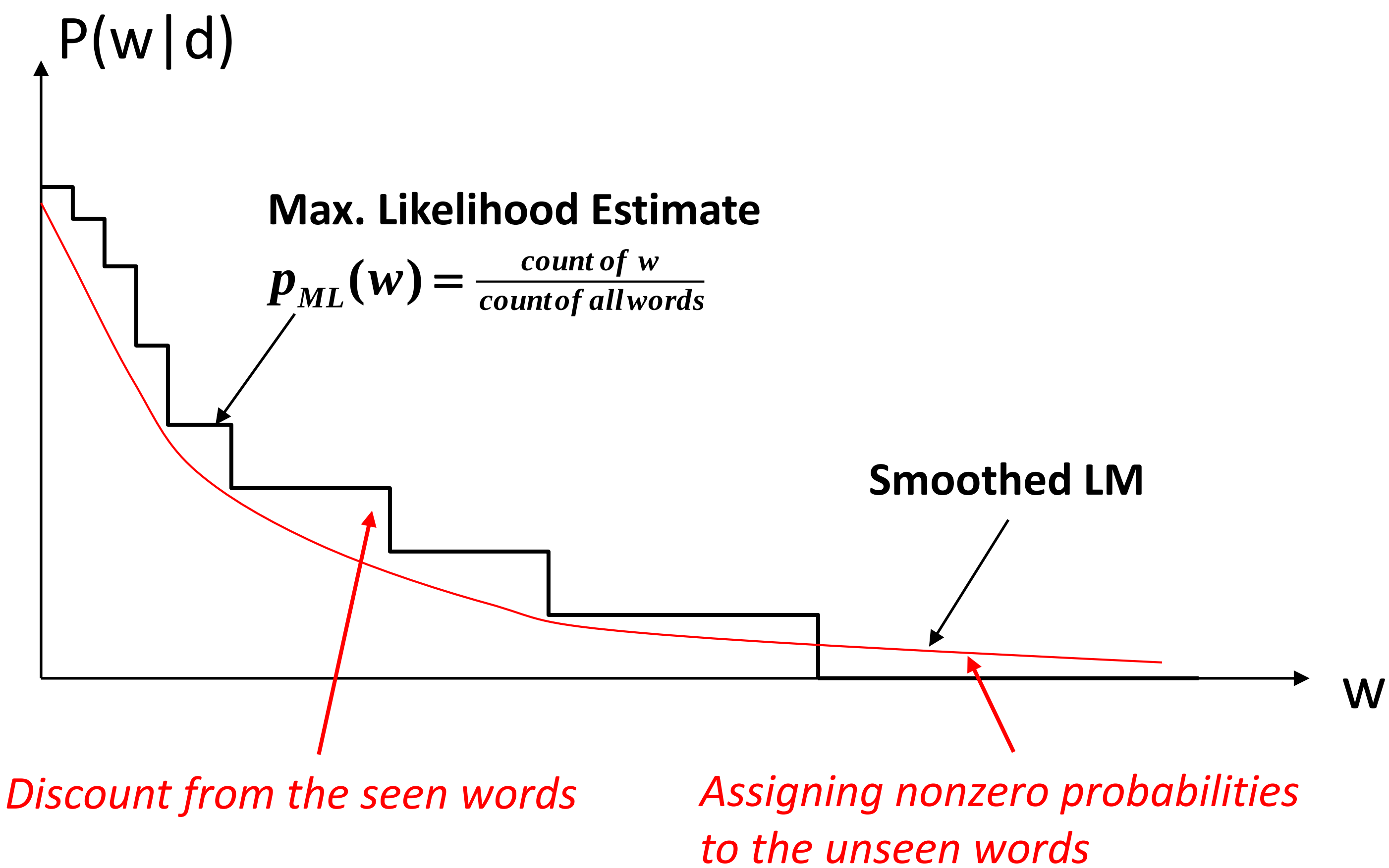
Document language model

– Retrieval problem $\approx$ Estimation of $p(w_i|d)$

– Common approach

  - Maximum likelihood estimation (MLE)

Recap: illustration of language model smoothing



Recap: smoothing methods

- Method 1: Additive smoothing
  - Add a constant δ to the counts of each word

Counts of w in d

$$p(w|d)=\frac{c(w,d)+1}{|d|+|V|}$$

“Add one”, Laplace smoothing

Vocabulary size

Length of d (total counts)

  - Problems?
    - Hint: all words are equally important?

Recap: refine the idea of smoothing

- Should all unseen words get equal probabilities?
- We can use a reference model to discriminate unseen words

Discounted ML estimate

$$p(w|d) = \begin{cases} p_{seen}(w|d) & \text{if } w \text{ is seen in } d \\ \alpha_d \, p(w|REF) & \text{otherwise} \end{cases}$$

Reference language model

$$\alpha_d = \frac{1 - \sum_{w \text{ is seen}} p_{seen}(w|d)}{\sum_{w \text{ is unseen}} p(w|REF)}$$

Recap: smoothing methods

- Method 2: Absolute discounting
  - Subtract a constant δ from the counts of each word

uniq words

$$p(w|d) = \frac{\max(c(w;d)-\delta,0)+\delta|d|_u \, p(w|REF)}{|d|}$$

  - Problems?
    - Hint: varied document length?

Recap: smoothing methods

- Method 3: Linear interpolation, Jelinek-Mercer
  - "Shrink" uniformly toward p(w|REF)

$$p(w|d) = (1-\lambda)\underbrace{\boxed{\frac{c(w,d)}{|d|}}}_{\text{MLE}} + \underbrace{\lambda}_{\text{parameter}}\, p(w|REF)$$

  - Problems?
    - Hint: what is missing?

Recap: smoothing methods

- Method 4: Dirichlet Prior/Bayesian
  - Assume pseudo counts $\mu p(w|REF)$

$$p(w|d) = \frac{c(w;d) + \mu p(w|REF)}{|d| + \mu} = \frac{|d|}{|d| + \mu} \frac{c(w,d)}{|d|} + \frac{\mu}{|d| + \mu} p(w|REF)$$

parameter

  - Problems?

Recap: understanding smoothing

Topical words

Query	**= "the**	**algorithms**	**for**	**data**	**mining"**	
p_{ML}(w|d1):	***0.04***	***0.001***	***0.02***	***0.002***	***0.003***	4.8×10^{-12}
p_{ML}(w|d2):	***0.02***	***0.001***	***0.01***	***0.003***	***0.004***	2.4×10^{-12}

p("algorithms"|d1) = p("algorithms"|d2)
p("data"|d1) < p("data"|d2)
p("mining"|d1) < p("mining"|d2)

Intuitively, d2 should have a higher score, but p(q|d1)>p(q|d2)…

So we should make p("the") and p("for") less different for all docs, and smoothing helps to achieve this goal…

2.35×10^{-13}

4.53×10^{-13}

After smoothing with $p(w|d) = 0.1 p_{DML}(w|d) + 0.9 p(w|REF)$, $p(q|d1) < p(q|d2)$!

Query	**= "the**	**algorithms**	**for**	**data**	**mining"**
P(w|REF)	**0.2**	**0.00001**	**0.2**	**0.00001**	**0.00001**
Smoothed p(w|d1):	**0.184**	**0.000109**	**0.182**	**0.000209**	**0.000309**
Smoothed p(w|d2):	**0.182**	**0.000109**	**0.181**	**0.000309**	**0.000409**

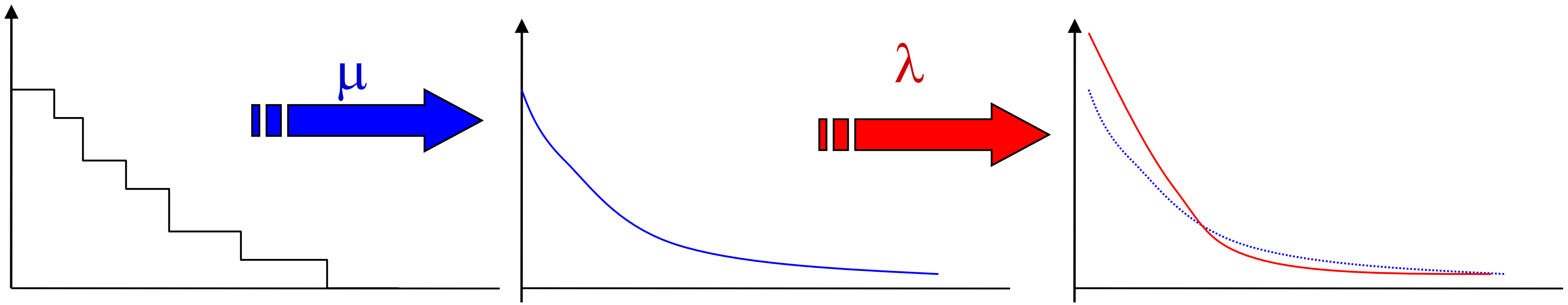
Two-stage smoothing [Zhai & Lafferty 02]

Stage-1

-Explain unseen words
-Dirichlet prior (Bayesian)

Stage-2

-Explain noise in query
-2-component mixture



Collection LM

$$P(w|d) = (1-\lambda)\frac{c(w,d) + \mu p(w|C)}{|d| + \mu} + \lambda p(w|U)$$

User background model

Understanding smoothing

$$\alpha_d = \frac{1 - \sum_{w \text{ is seen}} p_{seen}(w|d)}{\sum_{w \text{ is unseen}} p(w|REF)}$$

- Plug in the general smoothing scheme to the query likelihood retrieval formula, we obtain

TF weighting

Doc length normalization
(longer doc is expected to have a smaller α_d)

$$\log p(q|d) = \sum_{w_i \in d \cap q} [\log \frac{p_{seen}(w_i|d)}{\alpha_d \; p(w_i|C)}] + |q| \log \alpha_d + \boxed{\sum_{w_i \in q} \log p(w_i|C)}$$

IDF weighting

Ignore for ranking

- **Smoothing with *p(w|C)* ≈ TF-IDF + doc-length normalization**

Smoothing & TF-IDF weighting

Smoothed ML estimate

Retrieval formula using the general smoothing scheme

$$p(w|d) = \begin{cases} p_{Seen}(w|d) & if \;\; w \;\; is \;\; seen \; in \; d \\ \alpha_d \, p(w|C) & otherwise \end{cases}$$

$$\alpha_d = \frac{1 - \sum_{w \text{ is seen}} p_{Seen}(w|d)}{\sum_{w \text{ is unseen}} p(w|C)}$$

Reference language model

$$\log p(q|d) = \sum_{w \in V, c(w,q)>0} c(w,q) \log p(w|d)$$

Key rewriting step (where did we see it before?)

Similar rewritings are very common when using probabilistic models for IR…

What you should know

- How to estimate a language model
- General idea and different ways of smoothing
- Effect of smoothing

Today's reading

- Introduction to information retrieval
  - Chapter 12: Language models for information retrieval