Language Models

Slides borrowed from Hongning Wang with modifications

Query likelihood language models

$P(d/q) = P(q/d)*P(d)/P(q)$

How likely can the query be generated given a language model built based on the document?

Prior probability of a document, often treated as uniform across all documents, so can be ignored.

The same for all documents.

Generally involves two steps:

(1) estimate a language model based on D

(2) compute the query likelihood according to the estimated model

Language models!

What is a statistical LM?

- A model specifying a probability distribution over word <u>sequences</u>
  - 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 is 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)

Language models

We need independence assumptions!

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

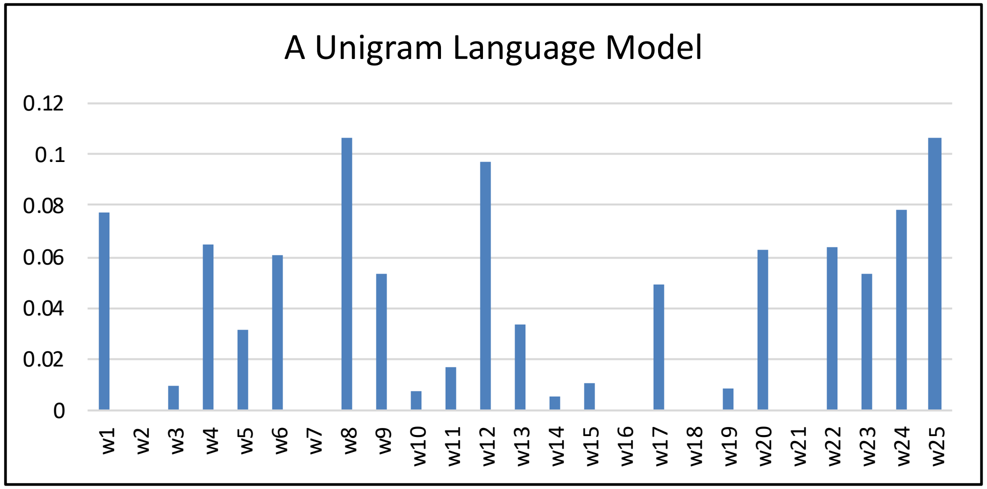
Chain rule: from conditional probability to joint probability

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

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!

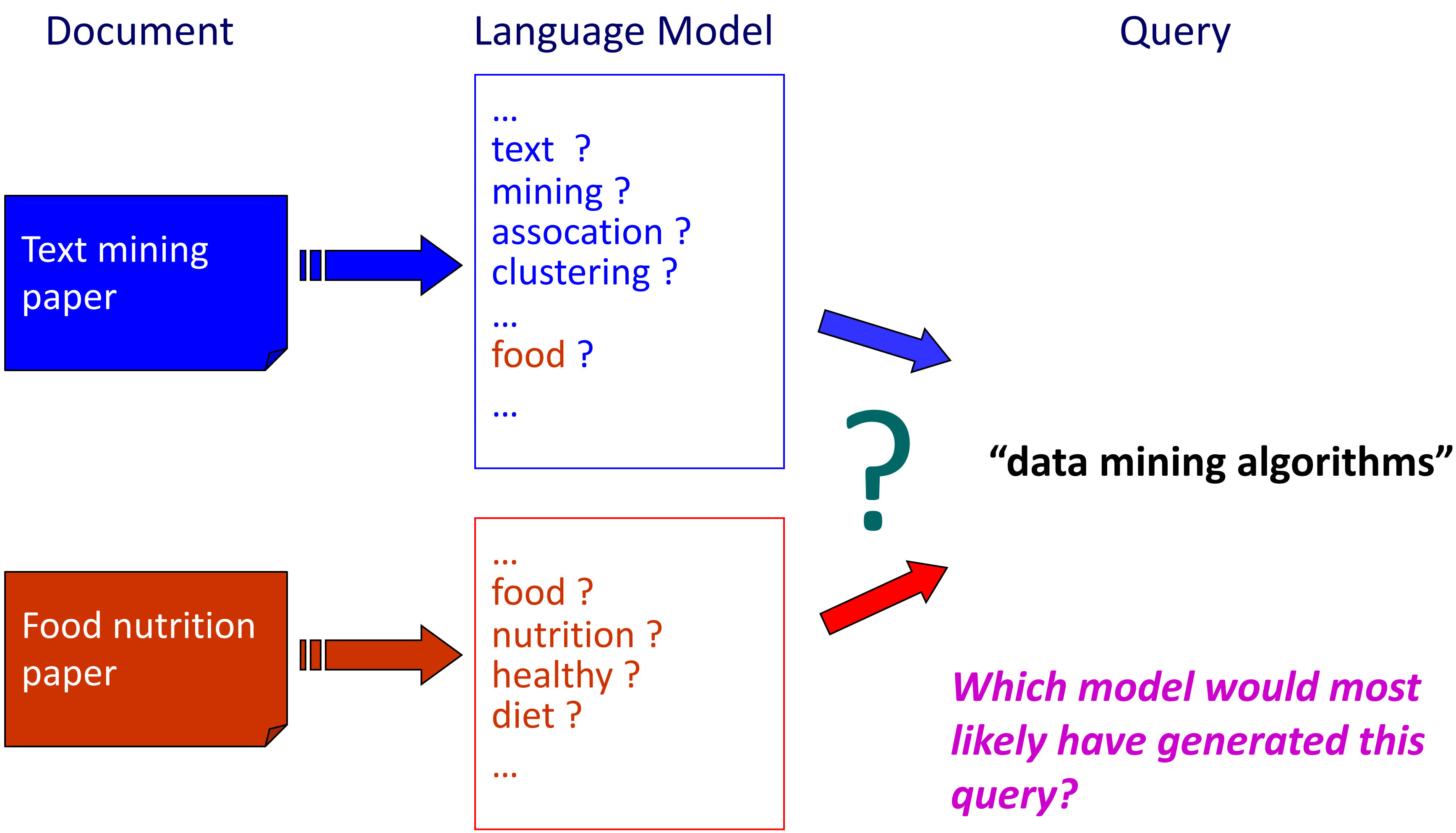
Higher-order LMs

- N-gram language models
  - In general, $p(w_1\, w_2\ \ldots w_n) = p(w_1)p(w_2|w_1) \ldots p(w_n|w_1 \ldots w_{n-1})$
  - N-gram: conditioned only on the past N-1 words
  - E.g., bigram: $p(w_1\ \ldots\ w_n) = p(w_1)p(w_2|w_1)\, p(w_3|w_2)\ \ldots p(w_n|w_{n-1})$

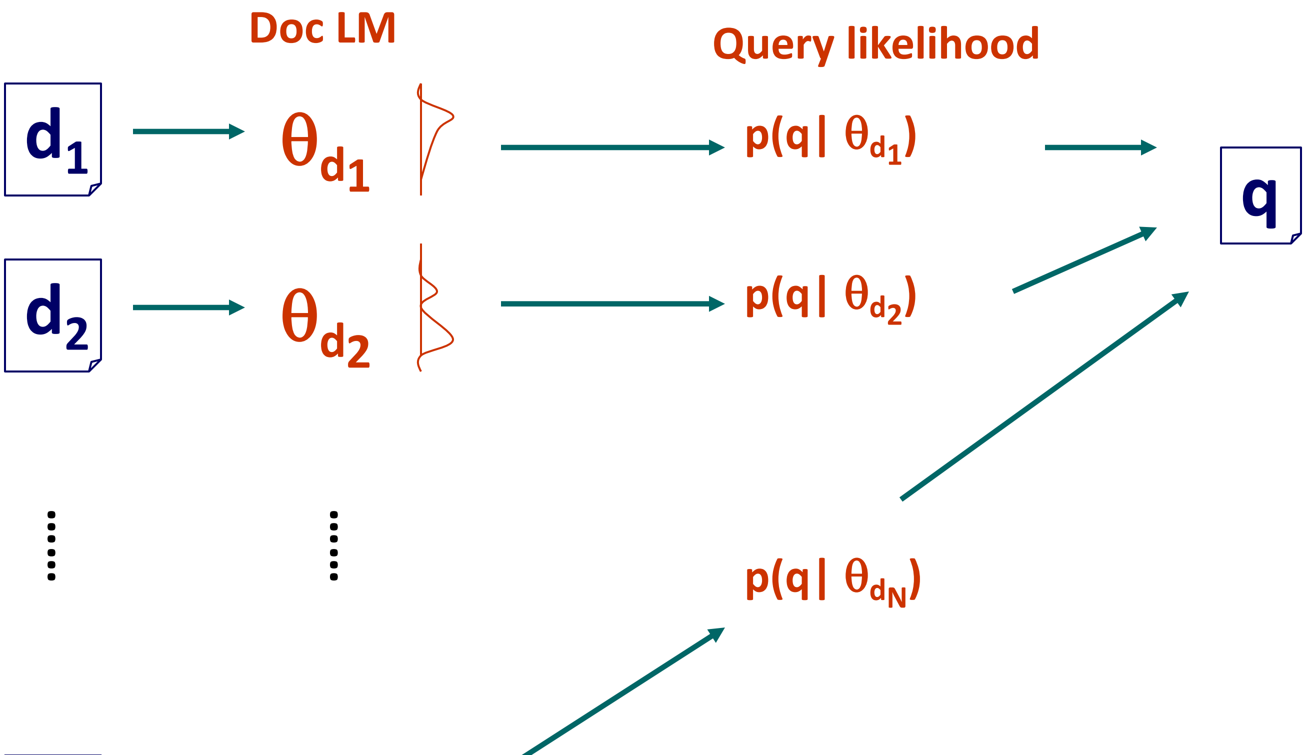
Why just unigram models?

- 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 ...

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



Ranking docs by query likelihood



Justification: PRP

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

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

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

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

maximum likelihood estimation

- Data: a document d with counts $c(w_1)$, …, $c(w_N)$
- Model: multinomial distribution $p(W|\theta)$ with parameters $\theta_i = p(w_i)$
- Maximum likelihood estimator: $\hat{\theta} = argmax_{\theta} p(W|\theta)$

$$p(W|\theta) = \binom{N}{c(w_1), \dots, c(w_N)} \prod_{i=1}^{N} \theta_i^{c(w_i)} \propto \prod_{i=1}^{N} \theta_i^{c(w_i)} \Rightarrow \log p(W|\theta) = \sum_{i=1}^{N} c(w_i) \log \theta_i$$

$$\Rightarrow L(W,\theta) = \sum_{i=1}^{N} c(w_i) \log \theta_i + \lambda \left(\sum_{i=1}^{N} \theta_i - 1 \right)$$

Using Lagrange multiplier approach, we'll tune $\boldsymbol{\theta_i}$ to maximize $\boldsymbol{L(W,\theta)}$

$$\Rightarrow \frac{\partial L}{\partial \theta_i} = \frac{c(w_i)}{\theta_i} + \lambda \quad \rightarrow \quad \theta_i = -\frac{c(w_i)}{\lambda}$$

Set partial derivatives to zero

$\Rightarrow$ Since $\sum_{i=1}^{N} \theta_i = 1$ we have $\lambda = -\sum_{i=1}^{N} c(w_i)$

Requirement from probability

$$\Rightarrow \theta_i = \boxed{\frac{c(w_i)}{\sum_{i=1}^{N} c(w_i)}}$$

ML estimate

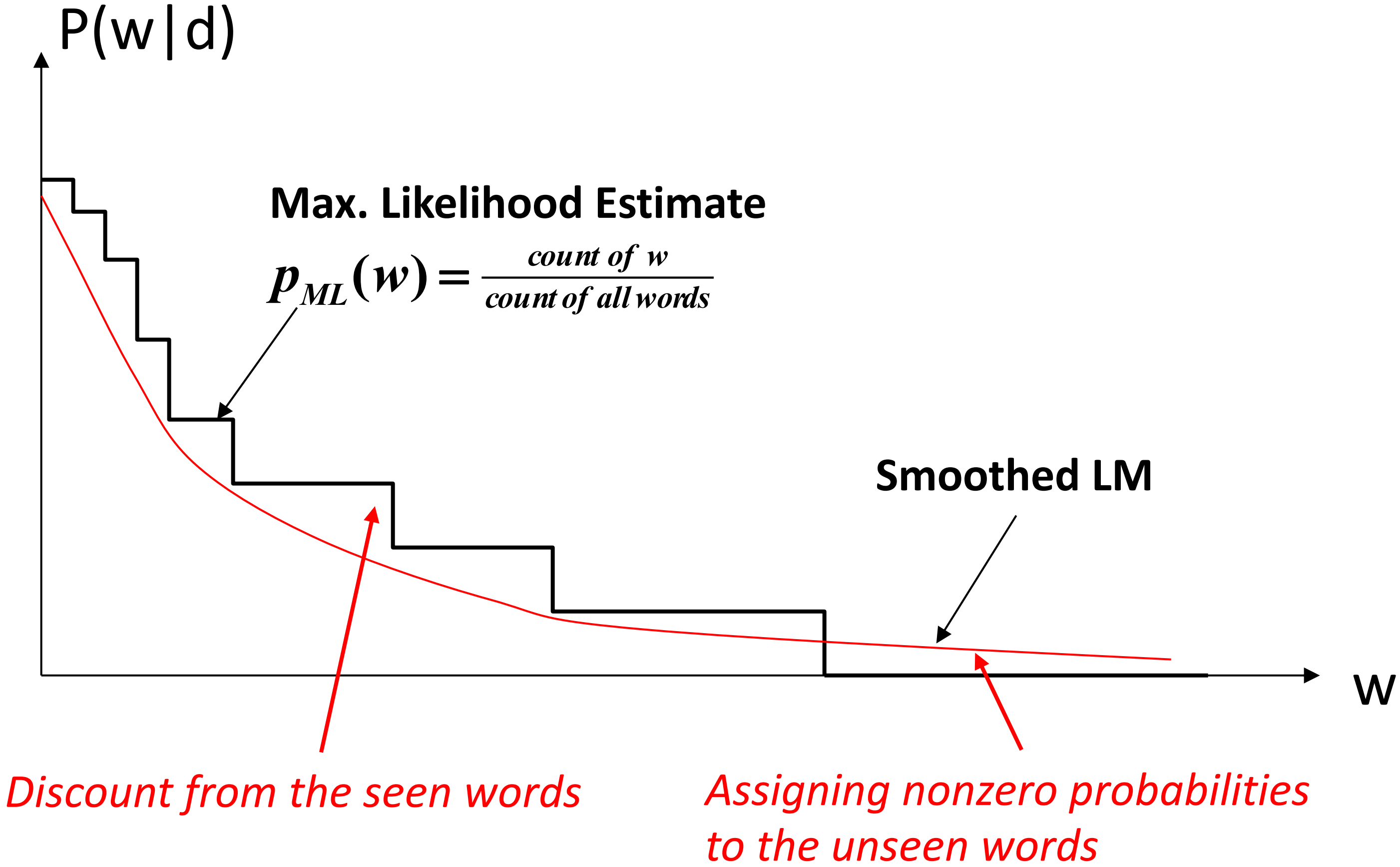
Problem with MLE

- 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



What you should know

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

Today's reading

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