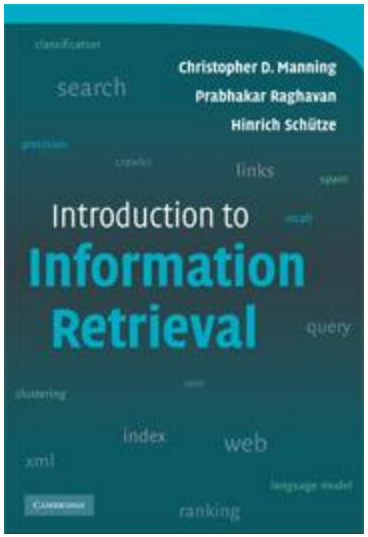
Information Retrieval and Organisation

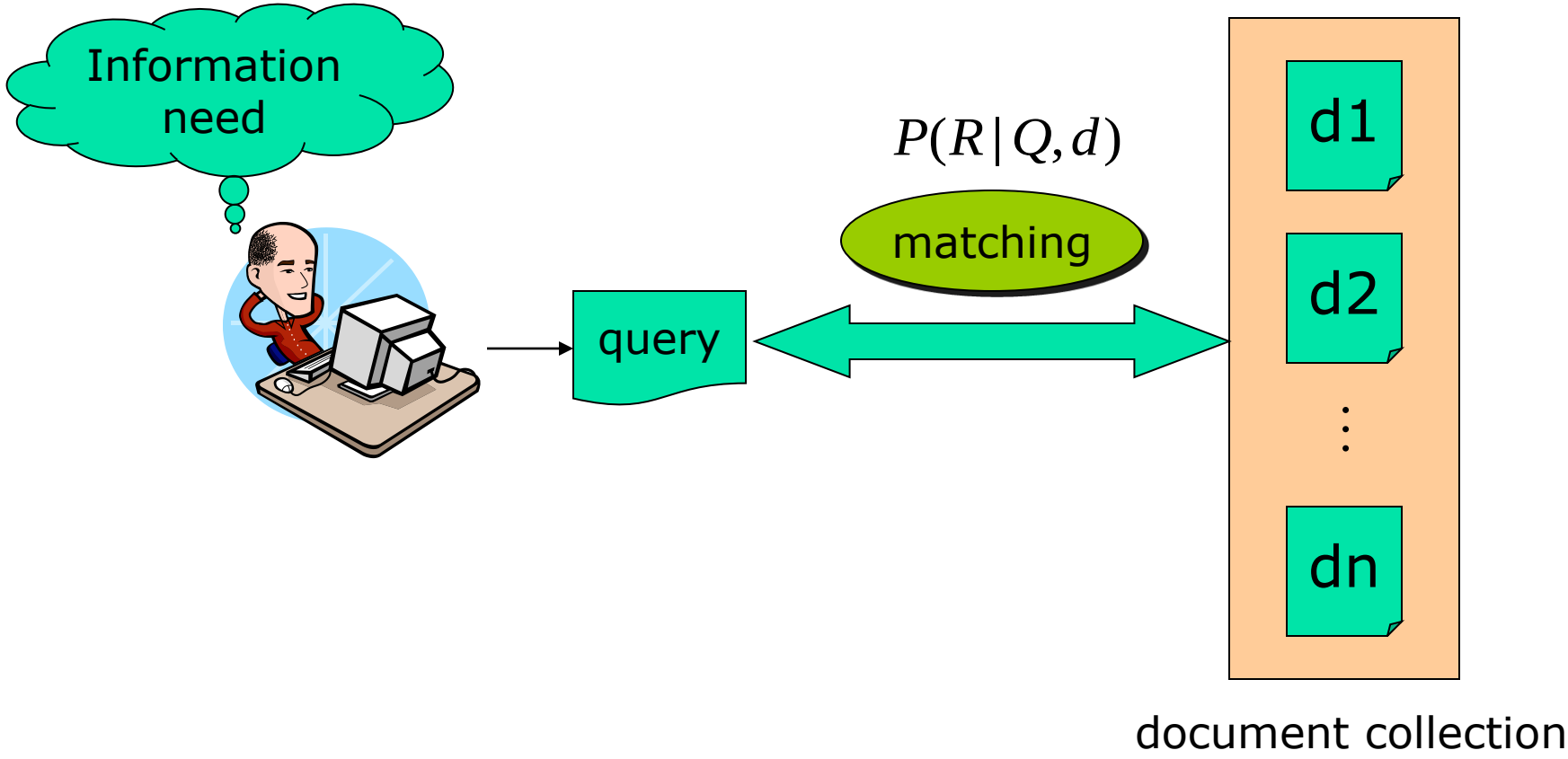


Chapter 12

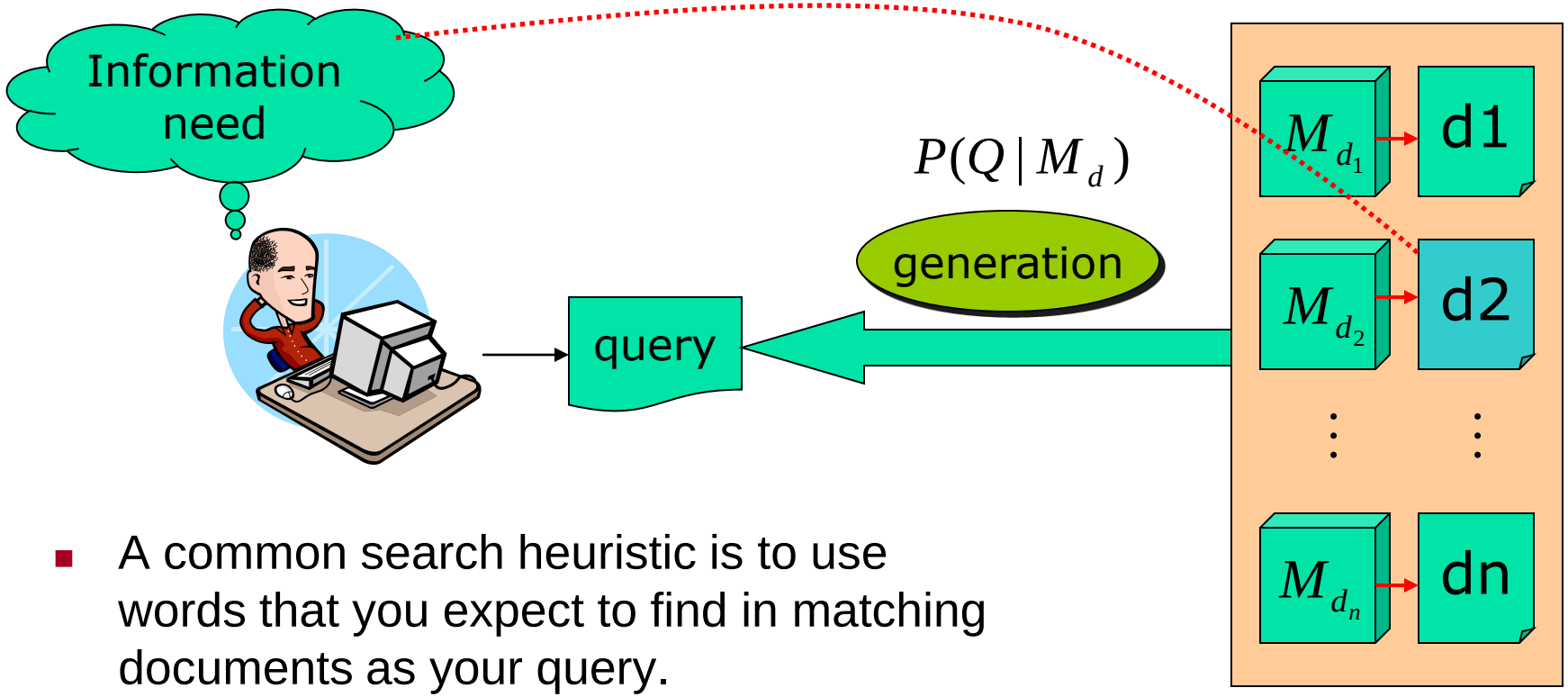
Language Models for Information Retrieval

Dell Zhang
Birkbeck, University of London

Standard Probabilistic IR



IR based on Language Models (LM)



- A common search heuristic is to use words that you expect to find in matching documents as your query.
- The LM approach directly exploits that idea!

What is a Statistical LM

- A probability distribution over word sequences
  - $P(\text{“}\textit{Today is Wednesday}\text{”}) \approx 0.001$
  - $P(\text{“}\textit{Today Wednesday is}\text{”}) \approx 0.0000000000001$
  - $P(\text{“}\textit{The eigenvalue is positive”}) \approx 0.00001$
- Can also be regarded as a probabilistic mechanism for “generating” text, thus also called a “generative” model
  - A piece of text can be regarded as a *sample* drawn according to the word distribution
- Context/Topic dependent!

Why is a LM Useful

- 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?
  - Given that we observe "*baseball*" three times and "*game*" once in a news article, how likely is it about sports?
  - Given that a user is interested in sports news, how likely would the user use "*baseball*" in a query?

The Simplest Language Model

- Unigram Model
  - Generate a piece of text by generating each word independently
  - Thus, $P(t_1\ t_2\ \ldots\ t_n) = P(t_1)\ P(t_2)\ \ldots\ P(t_n)$
  - Essentially a multinomial distribution over words

$$P_{\text{uni}}(t_1 t_2 t_3 t_4) = P(t_1)P(t_2)P(t_3)P(t_4)$$

Unigram LM

Model *M*

0.2	the
0.1	a
0.01	man
0.01	woman
0.03	said
0.02	likes
…	

the	man	likes	the	woman
0.2	0.01	0.02	0.2	0.01

multiply

$P(s|M) = 0.00000008$

Unigram LM

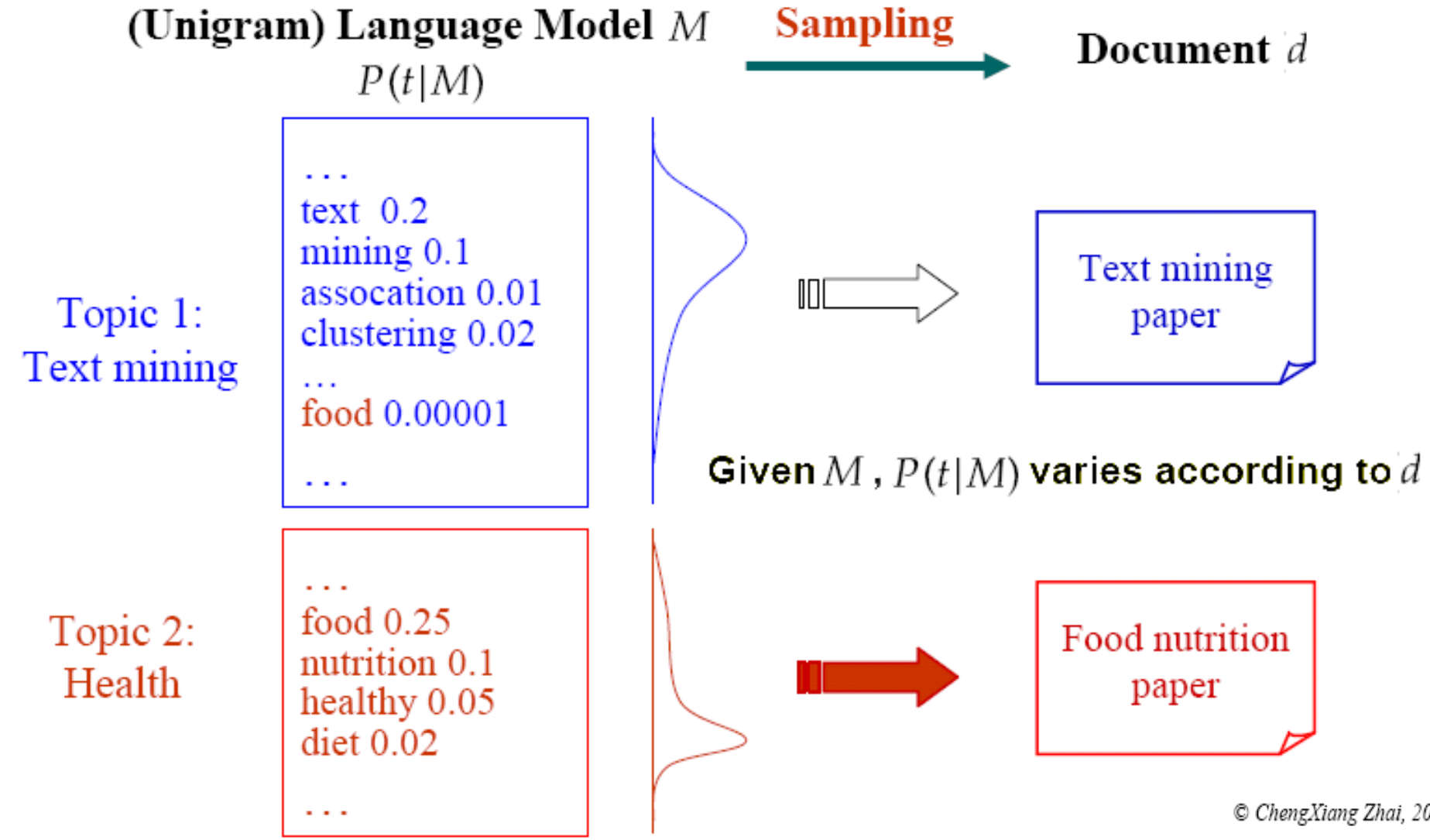
Model M_1	
0.2	the
0.01	class
0.0001	sayst
0.0001	pleaseth
0.0001	yon
0.0005	maiden
0.01	woman

Model M_2	
0.2	the
0.0001	class
0.03	sayst
0.02	pleaseth
0.1	yon
0.01	maiden
0.0001	woman

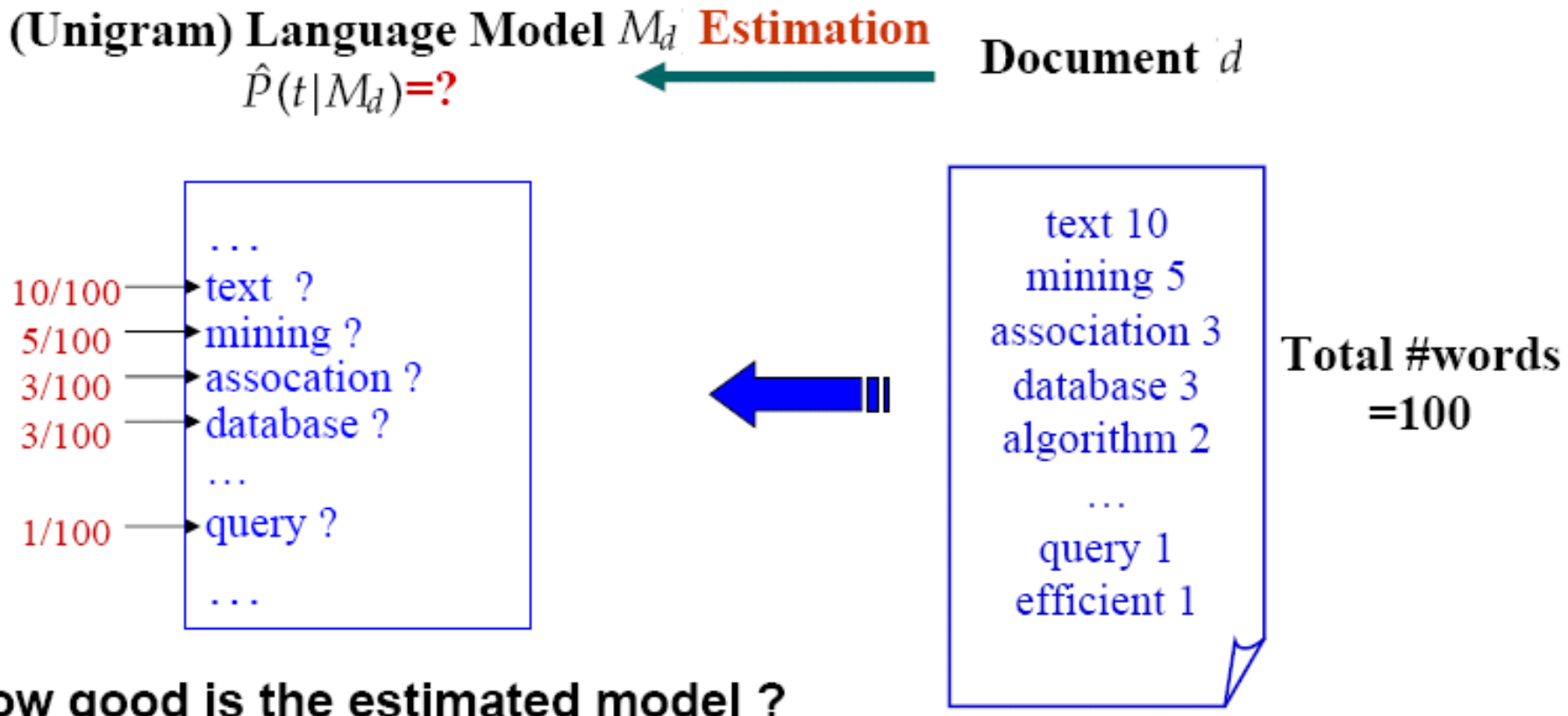
the	class	pleaseth	yon	maiden
0.2	0.01	0.0001	0.0001	0.0005
0.2	0.0001	0.02	0.1	0.01

$$P(s|M_2) > P(s|M_1)$$

Text Generation with Unigram LM



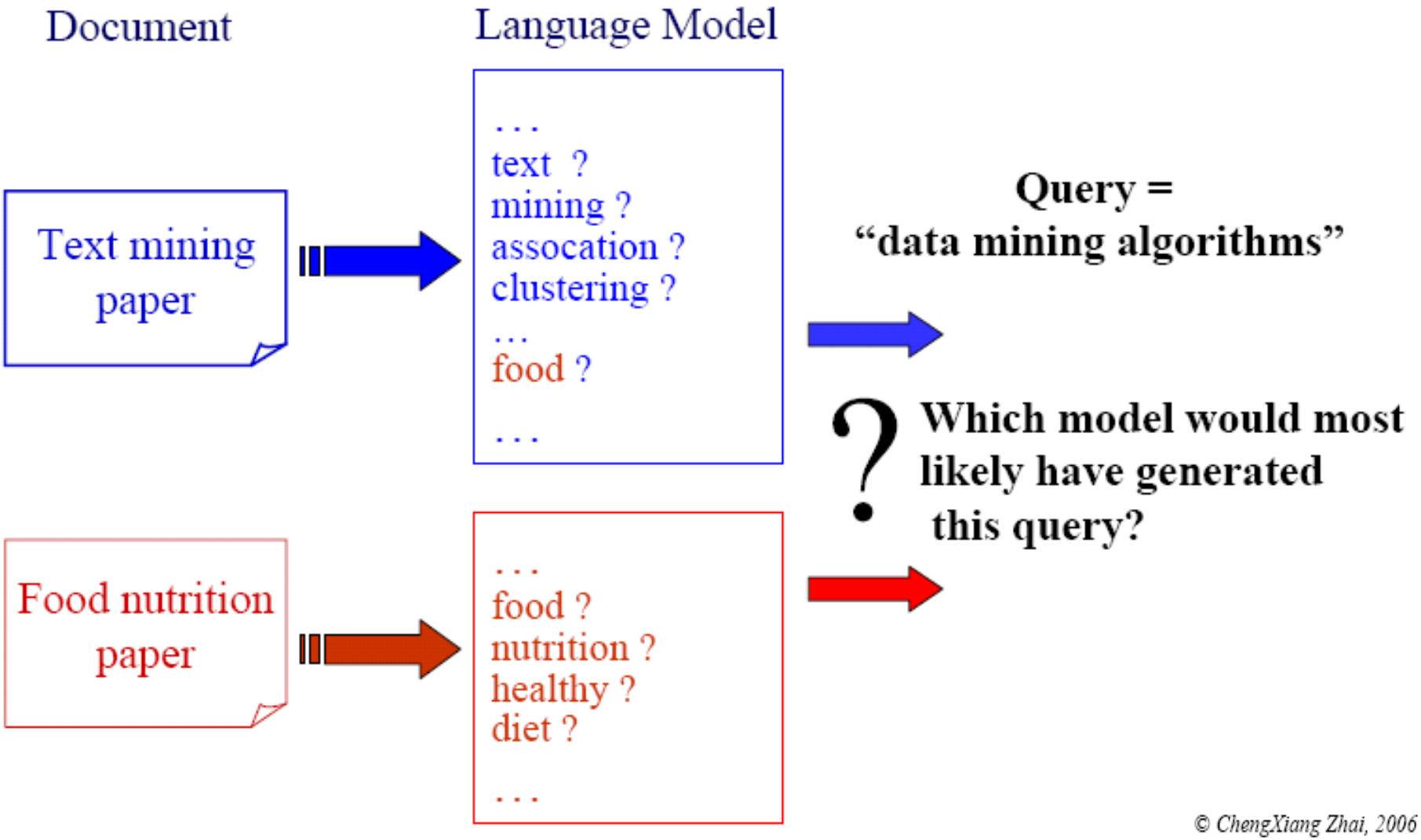
Estimation of Unigram LM



How good is the estimated model ?

It gives our document sample the highest prob,
but it doesn't generalize well… More about this later…

Using LMs for IR

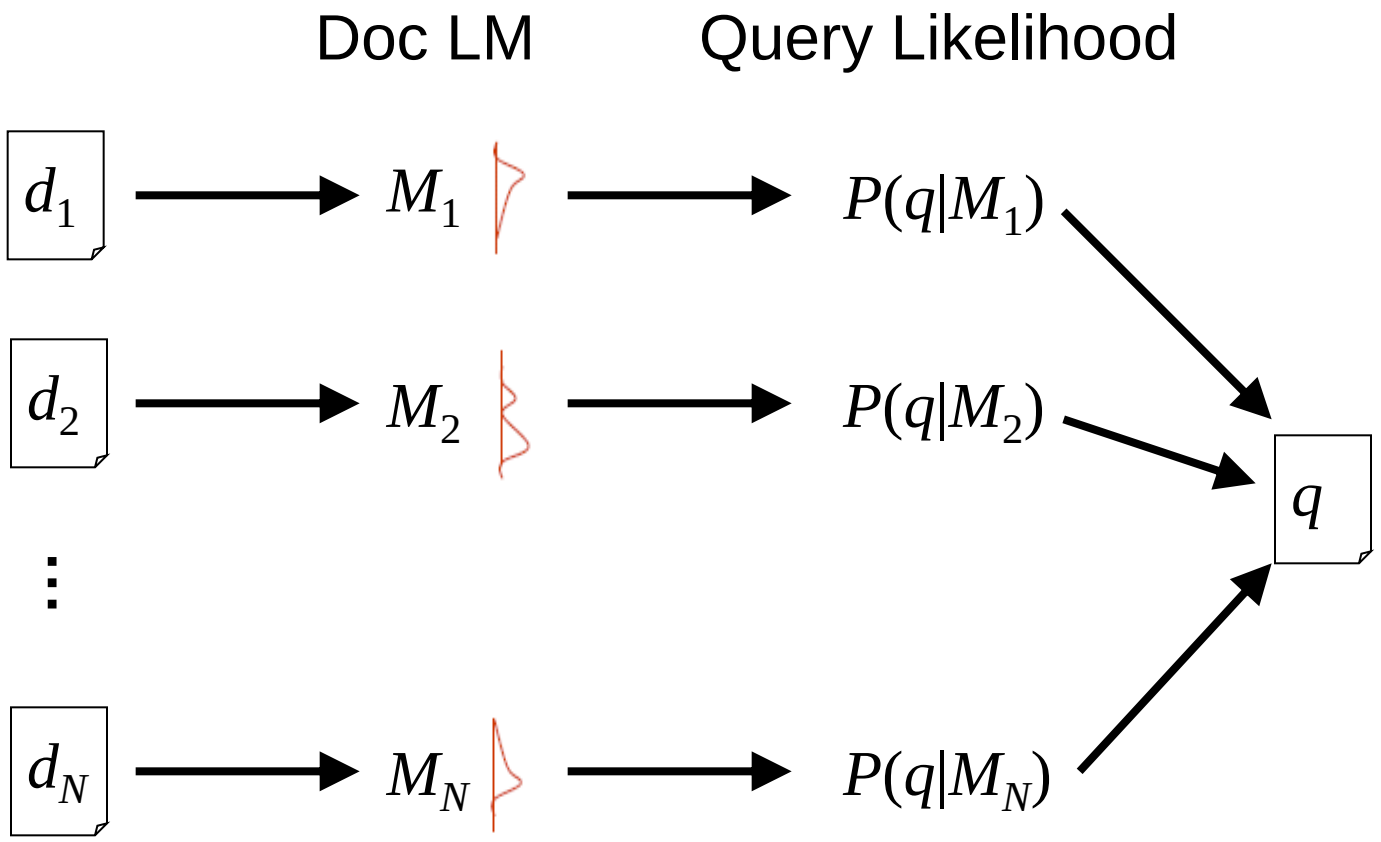


Using LMs for IR

- Relevance of a document: $P(d|q)$
- Using Bayes rule: $P(d|q) = P(q|d)\,P(d) / P(q)$
  - $P(q)$ is the same for all documents, so ignore
  - $P(d)$ [the prior] is often treated as uniform for all d
    - But we could use criteria like authority, length, genre, etc.
  - $P(q|d)$ [the likelihood] is the probability of q under d's language model M_d

Using LMs for IR

- Rank documents by query likelihood



Using LMs for IR

- It actually models the **query generation process**
  - Documents are ranked by the probability that a query would be observed as a random sample from the respective document model. $P(q|M_{d_i})$
  - Intuition: the user has a prototype document in mind and generates a query based on words that appear in this document. Often, users have a reasonable idea of terms that are likely to occur in documents of interest and they will choose query terms that distinguish these documents from others in the collection.

How to Construct LMs

- Simplest solution: Maximum Likelihood Estimation (MLE)
  - $P(t|M_d)$ = relative frequency of word t in d
- Problems of MLE
  - What if a word doesn't appear in the text? $P(t|M_d) = 0$ would be problematic

$$\hat{P}(q|M_d) = \prod_{t \in q} \hat{P}_{\text{mle}}(t|M_d) = \prod_{t \in q} \frac{\text{tf}_{t,d}}{L_d}$$

Smoothing for LMs

- In general, what probability should we give a word that has not been observed?
  - If we want to assign non-zero probabilities to such words, we'll have to discount the probabilities of observed words
  - This is what "smoothing" is about …

Smoothing for LMs

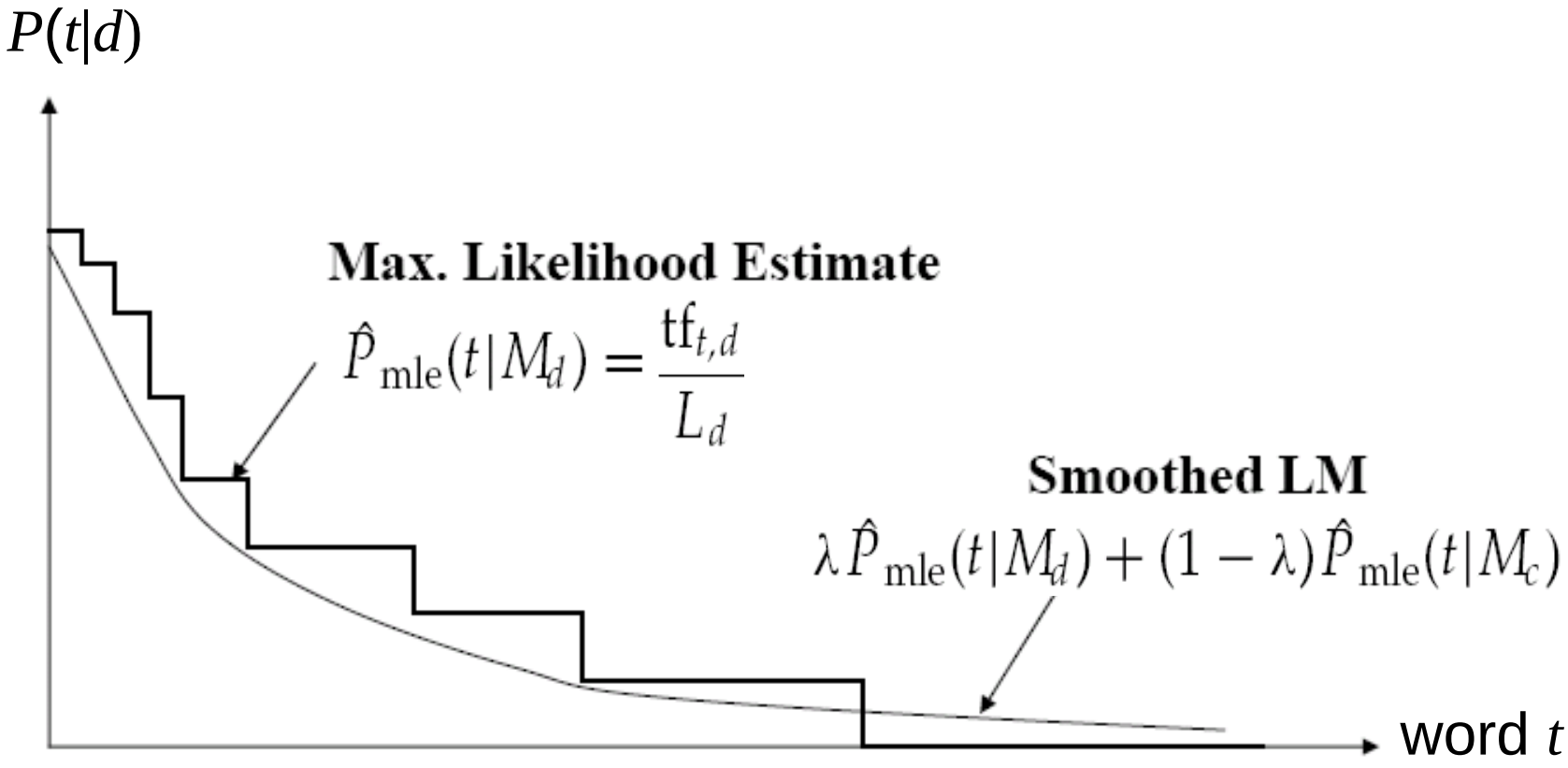
- Linear Interpolation LMs
  - A simple idea that works well in practice is to use a mixture between the document distribution and the collection distribution.
  - Correctly setting λ ($0< \lambda<1$) is very important to the good performance.

$$\hat{P}(t|d) = \lambda \hat{P}_{\text{mle}}(t|M_d) + (1-\lambda)\hat{P}_{\text{mle}}(t|M_c)$$

document model

collection model

Smoothing for LMs



Example

- Document Collection
  - d_1: Xerox reports a profit but revenue is down
  - d_2: Lucent narrows quarter loss but revenue decreases further
- Model
  - Unigram LM ($\lambda = ½$)
- Query:
  - q: revenue down

Example

- Ranking
  - $P(q|d_1) = [(1/8 + 2/16)/2] \times [(1/8 + 1/16)/2]$
    $= 1/8 \times 3/32 = 3/256$
  - $P(q|d_2) = [(1/8 + 2/16)/2] \times [(0 + 1/16)/2]$
    $= 1/8 \times 1/32 = 1/256$
- Results
  - $d_1 > d_2$

More Sophisticated LMs

- N-gram Models
  - In general, $P(t_1 t_2 t_3 t_4) = P(t_1)P(t_2|t_1)P(t_3|t_1 t_2)P(t_4|t_1 t_2 t_3)$
  - n-gram: conditioned only on the past n-1 words
  - e.g., bigram: $P_{\text{bi}}(t_1 t_2 t_3 t_4) = P(t_1)P(t_2|t_1)P(t_3|t_2)P(t_4|t_3)$
- 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 (A doc is an extremely small sample)
  - 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 ...

Appraisal

- Pros
  - A novel way of looking at the problem of IR based on probabilistic language modeling
    - mathematically precise
    - conceptually simple
    - computationally tractable
    - intuitively appealing
  - Importing the techniques from speech recognition and natural language processing
  - Often beats TFxIDF and BM25 in IR experiments

Appraisal

- Cons
  - The assumption of equivalence between document and query representation is unrealistic
  - Usually very simple models of language (e.g., unigram) is used
  - Without an explicit notion of relevance, relevance feedback is difficult to integrate into the model, as are user preferences
  - Can’t easily accommodate notions of phrase and passage matching or Boolean retrieval operators

Tools