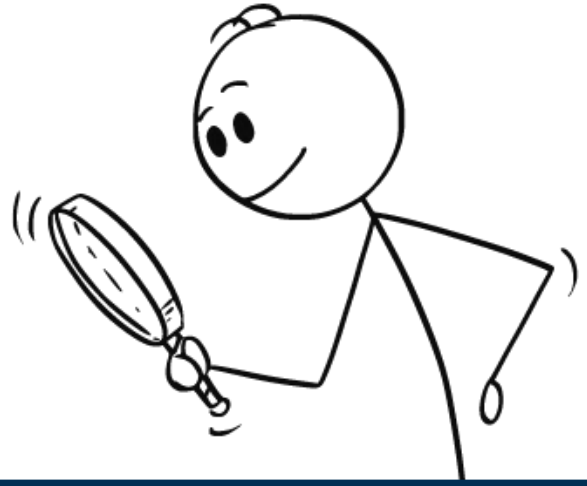


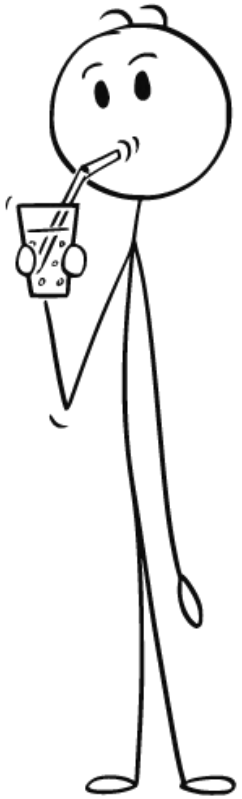
CSE 4/535
Information Retrieval

Sayantan Pal
PhD Student, Department of CSE
338Z Davis Hall

University at Buffalo
Department of CSE

Before we start

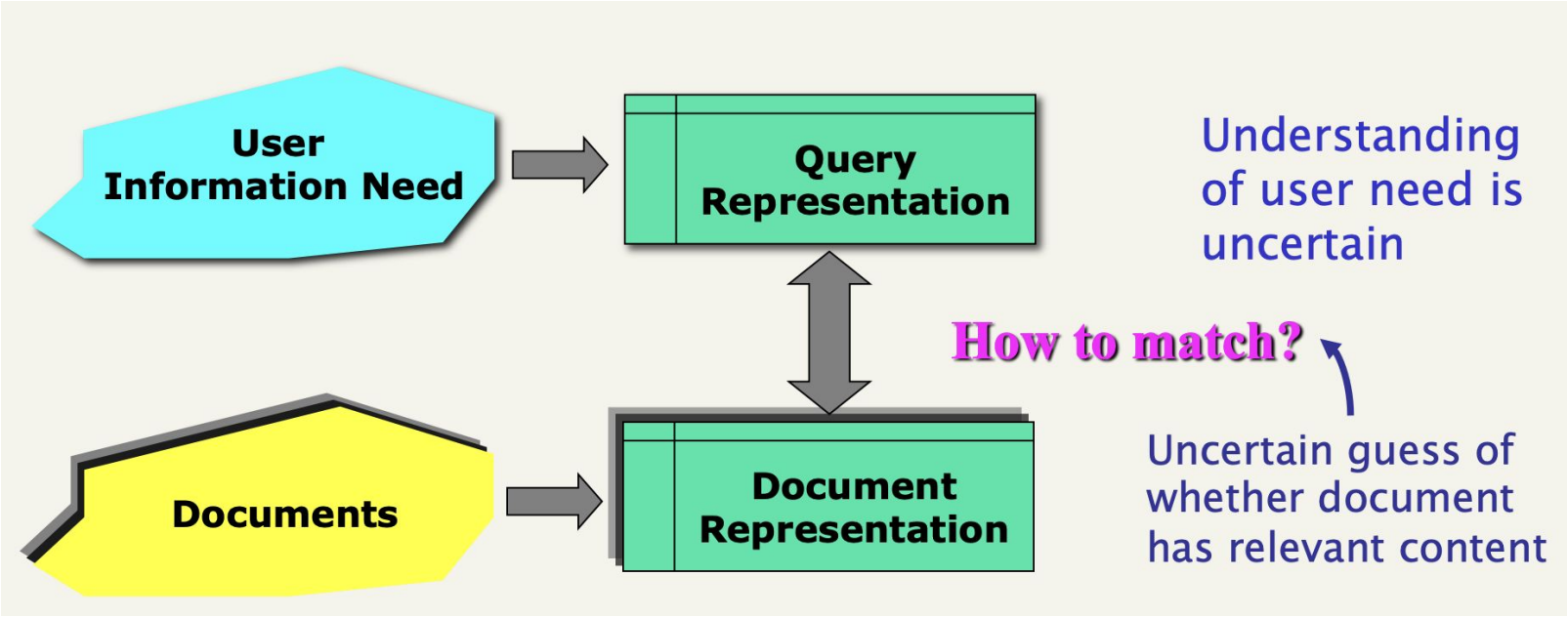
1. Midterm 2 is on 13th November (based on Poll)
2. Timeline is updated based on Poll responses, check website.
3. Final Project: Upto 3 members will be allowed
4. Today's lecture - Probabilistic Models in IR



Recap - Previous Class

1. Relevance Feedback
    a. Rocchio Algorithm
2. Query Expansion
    a. Word Co-occurrence

Why probabilities in IR?



- In traditional IR systems, matching between each document and query is attempted in a semantically imprecise space of index terms.
- Probabilities provide a principled foundation for uncertain reasoning. Can we use probabilities to quantify our uncertainties?

The document ranking problem

- We have a collection of documents
- User issues a query
- A list of documents needs to be returned
- Ranking method is core of an IR system:
  - In what order do we present documents to the user?
  - We want the "best" document to be first, second best second, etc….
- Idea: Rank by probability of relevance of the document w.r.t. information need
  - P(relevant|documenti , query)

Recall a few probability basics

- For events *a* and *b:*
- Bayes' Rule

$$p(a,b) = p(a \cap b) = p(a \mid b)p(b) = p(b \mid a)p(a)$$

$$p(\bar{a} \mid b) = p(b \mid \bar{a})p(\bar{a})$$

$$p(a \mid b) = \frac{p(b \mid a)p(a)}{p(b)} = \frac{p(b \mid a)p(a)}{\sum_{x=a,\bar{a}} p(b \mid x)p(x)}$$

Prior

Posterior

- Odds: $O(a) = \frac{p(a)}{p(\bar{a})} = \frac{p(a)}{1-p(a)}$

Start with prior prob p(a); Revise (posterior) based on evidence event b, in cases where a does or does not hold

The Probability Ranking Principle

"If a reference retrieval system's response to each request is a ranking of the documents in the collection in order of decreasing probability of relevance to the user who submitted the request, where the probabilities are estimated as accurately as possible on the basis of whatever data have been made available to the system for this purpose, the overall effectiveness of the system to its user will be the best that is obtainable on the basis of those data."

The Probability Ranking Principle

Let x be a document in the collection.
Let R represent **relevance** of a document w.r.t. given (fixed) query and let NR represent **non-relevance.**

R={0,1} vs. NR/R

Need to find $p(R|x)$ - probability that a document x is **relevant.**

$$p(R \mid x) = \frac{p(x \mid R)p(R)}{p(x)}$$

$$p(NR \mid x) = \frac{p(x \mid NR)p(NR)}{p(x)}$$

$p(R)$,$p(NR)$ - prior probability of retrieving a (non) relevant document

$$p(R \mid x) + p(NR \mid x) = 1$$

$p(x|R)$, $p(x|NR)$ - probability that if a relevant (non-relevant) document is retrieved, it is x.

The Probability Ranking Principle

- Simple case: no selection costs or other utility concerns that would differentially weight errors
- ***Bayes' Optimal Decision Rule*** (return a set, rather than a ranking)
  - ***x*** is **relevant** iff $p(R|x) > p(NR|x)$
- PRP in action: Rank all documents by $p(R|x)$

The Probability Ranking Principle

- More complex case: retrieval costs.
  - Let *d* be a document
  - *C* - cost of retrieval of relevant document
  - *C'* - cost of retrieval of non-relevant document
- Probability Ranking Principle: if

$$C \cdot p(R|d) + C' \cdot (1 - p(R|d)) \leq C \cdot p(R|d') + C' \cdot (1 - p(R|d'))$$

for all *d' not yet retrieved*, then *d* **is the next document to be retrieved**

- **We won't further consider loss/utility from now on**

Document d should be retrieved before document d' if the expected cost of retrieving d is less than or equal to the expected cost of retrieving d'. This approach ensures that the documents are retrieved in an order that **minimizes the overall expected retrieval cost**, considering both relevance and the associated retrieval costs.

Probability Ranking Principle

- How do we compute all those probabilities?
  - Do not know exact probabilities, have to use estimates
  - Binary Independence Retrieval (BIR) – which we discuss next– is the simplest model
- Questionable assumptions
  - "Relevance" of each document is independent of relevance of other documents.
    - Really, it's bad to keep on returning **duplicates**
  - Term independence
  - Boolean model of relevance
  - That one has a single step information need
    - Seeing a range of results might let user refine query

Probabilistic Retrieval Strategy

- Estimate how terms contribute to relevance
    - How do things like tf, df, and length influence your judgments about document relevance?
        - One answer is the Okapi formulae (S. Robertson)
- Combine to find document relevance probability
- Order documents by decreasing probability

Probabilistic Ranking

Basic concept:

"For a given query, if we know some documents that are relevant, terms that occur in those documents should be given greater weighting in searching for other relevant documents.

By making assumptions about the *distribution of terms* and applying Bayes Theorem, it is possible to derive weights theoretically."

Van Rijsbergen

Binary Independence Model

- Traditionally used in conjunction with PRP
- **"Binary" = Boolean**: documents are represented as binary incidence vectors of terms (cf. lecture 1):
  - $\vec{x} = (x_1, \ldots, x_n)$
  - $x_i = 1$ iff term *i* is present in document *x*.
- **"Independence":** terms occur in documents independently
- Different documents can be modeled as same vector

- Bernoulli Naive Bayes model (cf. text categorization!)
  - Equivalent to assumption of VSM where each term is a dimension orthogonal to all other terms

Binary Independence Model

- Queries: binary term incidence vectors
- Given query ***q***,
  - for each document ***d*** need to compute ***p(R|q,d).***
  - replace with computing ***p(R|q,x)*** where ***x*** is binary term incidence vector representing ***d*** Interested only in ranking
- Will use odds and Bayes' Rule:

$$O(R \mid q,\vec{x}) = \frac{p(R \mid q,\vec{x})}{p(NR \mid q,\vec{x})} = \frac{\frac{p(R \mid q)\,p(\vec{x} \mid R,q)}{p(\vec{x} \mid q)}}{\frac{p(NR \mid q)\,p(\vec{x} \mid NR,q)}{p(\vec{x} \mid q)}}$$

P(R | q) : probability of relevance for a particular query (same for all documents)

Binary Independence Model

$$O(R \mid q, \vec{x}) = \frac{p(R \mid q, \vec{x})}{p(NR \mid q, \vec{x})} = \frac{p(R \mid q)}{p(NR \mid q)} \cdot \frac{p(\vec{x} \mid R, q)}{p(\vec{x} \mid NR, q)}$$

Constant for a given query

Needs estimation

- Using **Independence** Assumption:

$$\frac{p(\vec{x} \mid R, q)}{p(\vec{x} \mid NR, q)} = \prod_{i=1}^{n} \frac{p(x_i \mid R, q)}{p(x_i \mid NR, q)}$$

- So : $$O(R \mid q, d) = O(R \mid q) \cdot \prod_{i=1}^{n} \frac{p(x_i \mid R, q)}{p(x_i \mid NR, q)}$$

Binary Independence Model

$$O(R \mid q,d) = O(R \mid q) \cdot \prod_{i=1}^{n} \frac{p(x_i \mid R,q)}{p(x_i \mid NR,q)}$$

- Since x_i is either 0 or 1:

$$O(R \mid q,d) = O(R \mid q) \cdot \prod_{x_i=1} \frac{p(x_i = 1 \mid R,q)}{p(x_i = 1 \mid NR,q)} \cdot \prod_{x_i=0} \frac{p(x_i = 0 \mid R,q)}{p(x_i = 0 \mid NR,q)}$$

- Let $p_i = p(x_i = 1 \mid R,q); \quad r_i = p(x_i = 1 \mid NR,q);$

probability that term i appears in a relevant document

Binary Independence Model

$$O(R \mid q,d) = O(R \mid q) \cdot \prod_{i=1}^{n} \frac{p(x_i \mid R,q)}{p(x_i \mid NR,q)}$$

- Since x_i is either 0 or 1:

$$O(R \mid q,d) = O(R \mid q) \cdot \prod_{x_i=1} \frac{p(x_i = 1 \mid R,q)}{p(x_i = 1 \mid NR,q)} \cdot \prod_{x_i=0} \frac{p(x_i = 0 \mid R,q)}{p(x_i = 0 \mid NR,q)}$$

- Let $p_i = p(x_i = 1 \mid R,q); \quad r_i = p(x_i = 1 \mid NR,q);$

probability that term i appears in a relevant document

- Assume, for all terms not occurring in the query (q_i=0) $p_i = r_i$

Then...

i.e. they are equally likely
To occur in R and NR docs
This can be
changed (e.g., in
relevance feedback)

Binary Independence Model

$$O(R \mid q, \vec{x}) = O(R \mid q) \cdot \prod_{x_i = q_i = 1} \frac{p_i}{r_i} \cdot \prod_{\substack{x_i = 0 \\ q_i = 1}} \frac{1 - p_i}{1 - r_i}$$

All matching terms

Non-matching query terms

$$= O(R \mid q) \cdot \prod_{x_i = q_i = 1} \frac{p_i (1 - r_i)}{r_i (1 - p_i)} \cdot \prod_{q_i = 1} \frac{1 - p_i}{1 - r_i}$$

All matching terms

All query terms

Step 1: Left product is over query terms found in doc; right product over query terms not in doc

Step 2: Manipulate expression: include query terms found in doc into right product, simultaneously divide through by them into left product

Binary Independence Model

$$O(R \mid q, \vec{x}) = O(R \mid q) \cdot \prod_{x_i = q_i = 1} \frac{p_i(1-r_i)}{r_i(1-p_i)} \cdot \prod_{q_i = 1} \frac{1-p_i}{1-r_i}$$

Constant for each query

Only quantity to be estimated for rankings

- Retrieval Status Value:

$$RSV = \log \prod_{x_i = q_i = 1} \frac{p_i(1-r_i)}{r_i(1-p_i)} = \sum_{x_i = q_i = 1} \log \frac{p_i(1-r_i)}{r_i(1-p_i)}$$

Binary Independence Model

- All boils down to computing RSV.

$$RSV = \log \prod_{x_i=q_i=1} \frac{p_i(1-r_i)}{r_i(1-p_i)} = \sum_{x_i=q_i=1} \log \frac{p_i(1-r_i)}{r_i(1-p_i)}$$

$$RSV = \sum_{x_i=q_i=1} c_i; \qquad c_i = \log \frac{p_i(1-r_i)}{r_i(1-p_i)}$$

Odds of term appearing if doc is Rel $\frac{p_i}{(1-p_i)}$ $\frac{r_i}{(1-r_i)}$ Odds of term appearing if doc is NonRel

Odds Ratio is ratio of these two odds; take log

Value is 0 if a term has equal odds of appearing in R and NR docs,
>0 if more likely to appear in R docs
So, how do we compute c_i's from our data ? **c_i's function as term weights**

Binary Independence Model

- Estimating RSV coefficients.
- For each term *i* look at this table of document counts:

Documens	Relevant	Non-Relevant	Total
$X_i=1$	s	$n-s$	n
$X_i=0$	$S-s$	$N-n-S+s$	$N-n$
Total	S	$N-S$	N

What is n?

- Estimates: $p_i \approx \frac{s}{S}$ $r_i \approx \frac{(n-s)}{(N-S)}$

$$c_i \approx K(N,n,S,s) = \log \frac{s/(S-s)}{(n-s)/(N-n-S+s)}$$

For now, assume no zero terms.

Smoothing

- In practice, do some smoothing to account for zero weights. Add .5 to all quantities.

$$\hat{c}_t = K(N, \mathrm{df}_t, S, s) = \log \frac{(s + \frac{1}{2})/(S - s + \frac{1}{2})}{(\mathrm{df}_t - s + \frac{1}{2})/(N - \mathrm{df}_t - S + s + \frac{1}{2})}$$

- what is a good estimate for # of non-rel docs?
- estimate of term occurrence in non-rel docs?
- what is a reasonable estimate for p_t?

Differences between TF-IDF and BIM weighting

- TF-IDF measures term frequency directly; BIM measures (estimated) proportion of relevant documents that the term t occurs in

$$c_t = \log\left[\frac{p_t}{1-p_t}\cdot\frac{1-u_t}{u_t}\right] \approx \log\left[\frac{|V_t|+\frac{1}{2}}{|V|-|V_t|+1}\cdot\frac{N}{\mathrm{df}_t}\right]$$

Looks sort of like tf-idf, but is not!

But things aren't quite the same: $p_t/(1-p_t)$ measures the (estimated) proportion of relevant documents that the term t occurs in, not term frequency. Moreover, if we apply log identities:

$$c_t = \log\frac{|V_t|+\frac{1}{2}}{|V|-|V_t|+1} + \log\frac{N}{\mathrm{df}_t}$$

Adding log-scaled components, not multiplying

- TF-IDF: words appearing in query but not in document have zero say in relevance
  BIM: counts their contribution by fraction of other such documents in collection (relevant, but do not contain this term)

Steps skipped to obtain the final BIM

- Probabilistic Relevance Feedback

Make sure to remember the difference

Okapi BM (Best Matching)-25: Non-Binary Model

- The BIM originally designed for short catalog records and abstracts of consistent length; works well in these contexts
- for modern full-text search collections, a model should pay attention to term frequency and document length
- The *BM25 weighting scheme* , *Okapi weighting* , developed as a way of building a probabilistic model sensitive to these quantities while not introducing too many additional parameters into the model (Sparck Jones et al., 2000)
- The simplest score for document is just idf weighting of the query terms present

$$RSV_d = \sum_{t \in q} \log \frac{N}{\mathrm{df}_t}$$

- in the absence of relevance feedback information we estimate that S-s=0 (absent terms in rel docs), then we get an alternative idf formulation as follows

$$RSV_d = \sum_{t \in q} \log \frac{N - \mathrm{df}_t + \frac{1}{2}}{\mathrm{df}_t + \frac{1}{2}}$$

If a term appears in more than half the docs, this generates negative weights; stop words solve this issue

Okapi BM (Best Matching)-25: Non-Binary Model

Recall RSV

- factoring in the frequency of each term and document length:

$$RSV_d = \sum_{t \in q} \log \left[\frac{N}{\mathrm{df}_t} \right] \cdot \frac{(k_1 + 1)\mathrm{tf}_{td}}{k_1((1 - b) + b \times (L_d / L_{ave})) + \mathrm{tf}_{td}}$$

- Where tf_{td} is tf of term t in doc d, L_d and L_{ave} are length of doc d and avg doc; k_1 is a positive tuning parameter that calibrates the doc tf scaling; k_1=0 is binary model; large value of k_1 is similar to using raw tf. b determines scaling by doc length; b=1 represents doc length scaling, b=0 represents no scaling

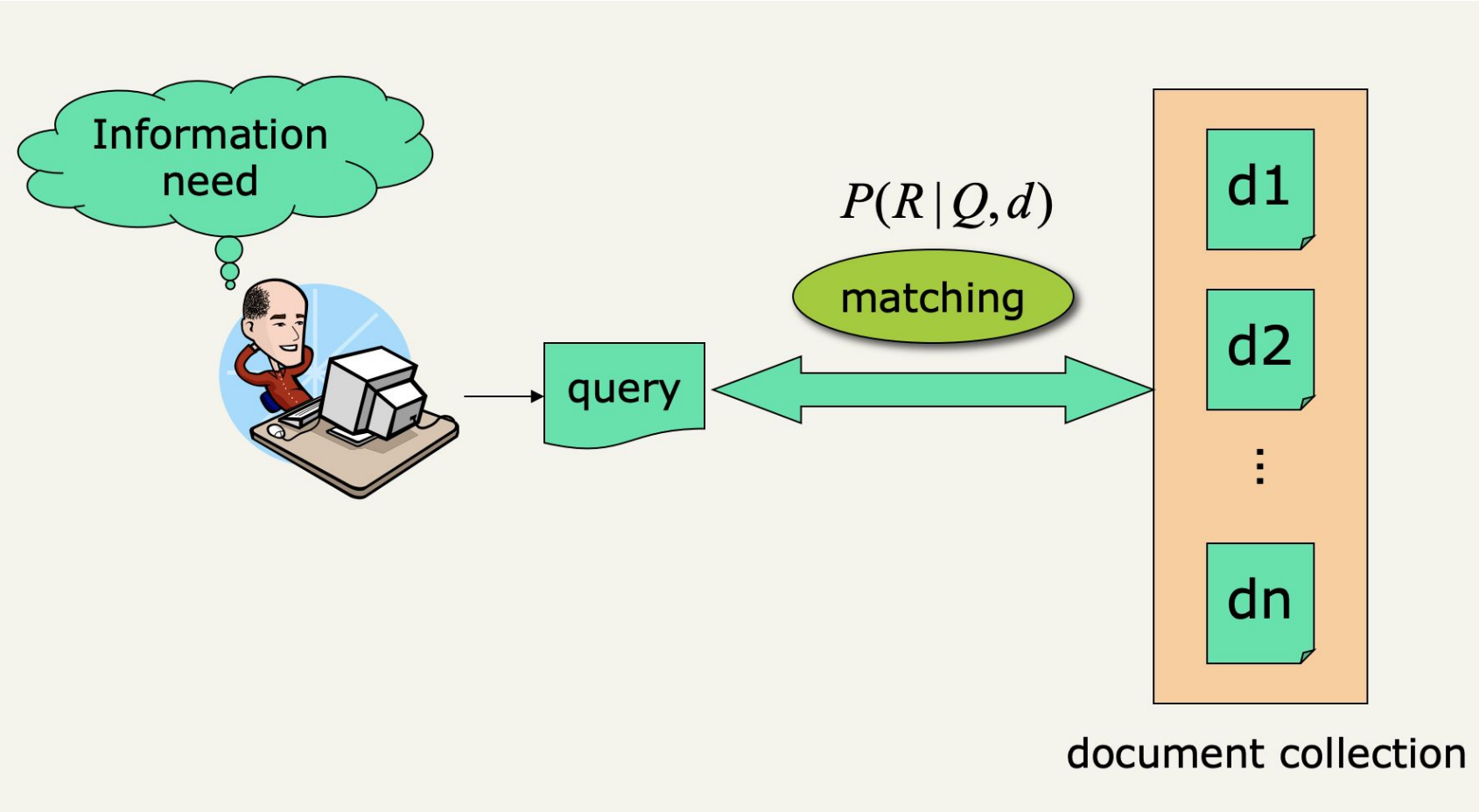
- If query is long, use similar weighting for query terms; appropriate if queries are paragraph long, unnecessary for short queries. k_3 scaling parameter for query frequency.

$$RSV_d = \sum_{t \in q} \left[\log \frac{N}{\mathrm{df}_t} \right] \cdot \frac{(k_1 + 1)\mathrm{tf}_{td}}{k_1((1 - b) + b \times (L_d / L_{ave})) + \mathrm{tf}_{td}} \cdot \frac{(k_3 + 1)\mathrm{tf}_{tq}}{k_3 + \mathrm{tf}_{tq}}$$

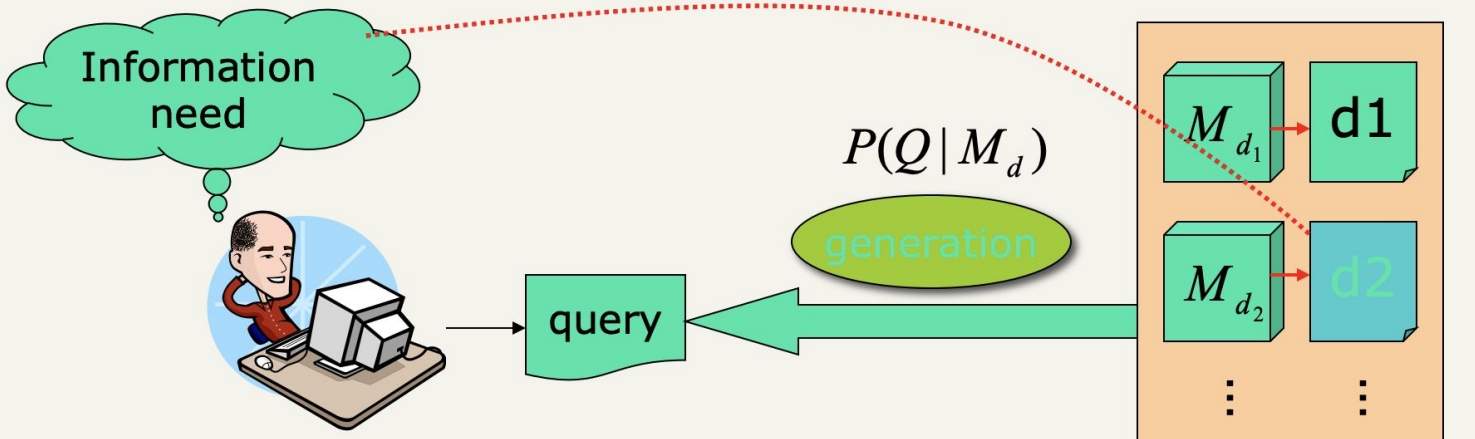
k1, k3 to a value between 1.2 and 2, b=0.75

Language Models

Standard Probabilistic IR



IR based on Language Model (LM)

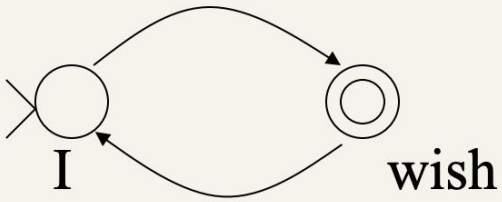


- A common search heuristic is to use words that you expect to find in matching documents as your query – why, I saw Sergey Brin advocating that strategy on late night TV one night in my hotel room, so it must be good!
- The LM approach directly exploits that idea!

What is the probability that the query was generated by a particular model?

Formal Language (Model)

- Traditional generative model: generates strings
  - Finite state machines or regular grammars, etc.
- Example:



I wish
I wish I wish
I wish I wish I wish
I wish I wish I wish I wish
…

*wish I wish

Stochastic Language Models

- Models *probability* of generating strings in the language (commonly all strings over alphabet ∑)

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 model

Stochastic Language Models

- Model *probability* of generating any string

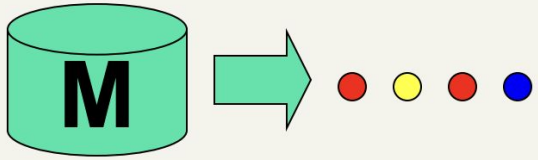
Model M1		Model M2	
0.2	the	0.2	the
0.01	class	0.0001	class
0.0001	sayst	0.03	sayst
0.0001	pleaseth	0.02	pleaseth
0.0001	yon	0.1	yon
0.0005	maiden	0.01	maiden
0.01	woman	0.0001	woman

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

$$P(s|M2) > P(s|M1)$$

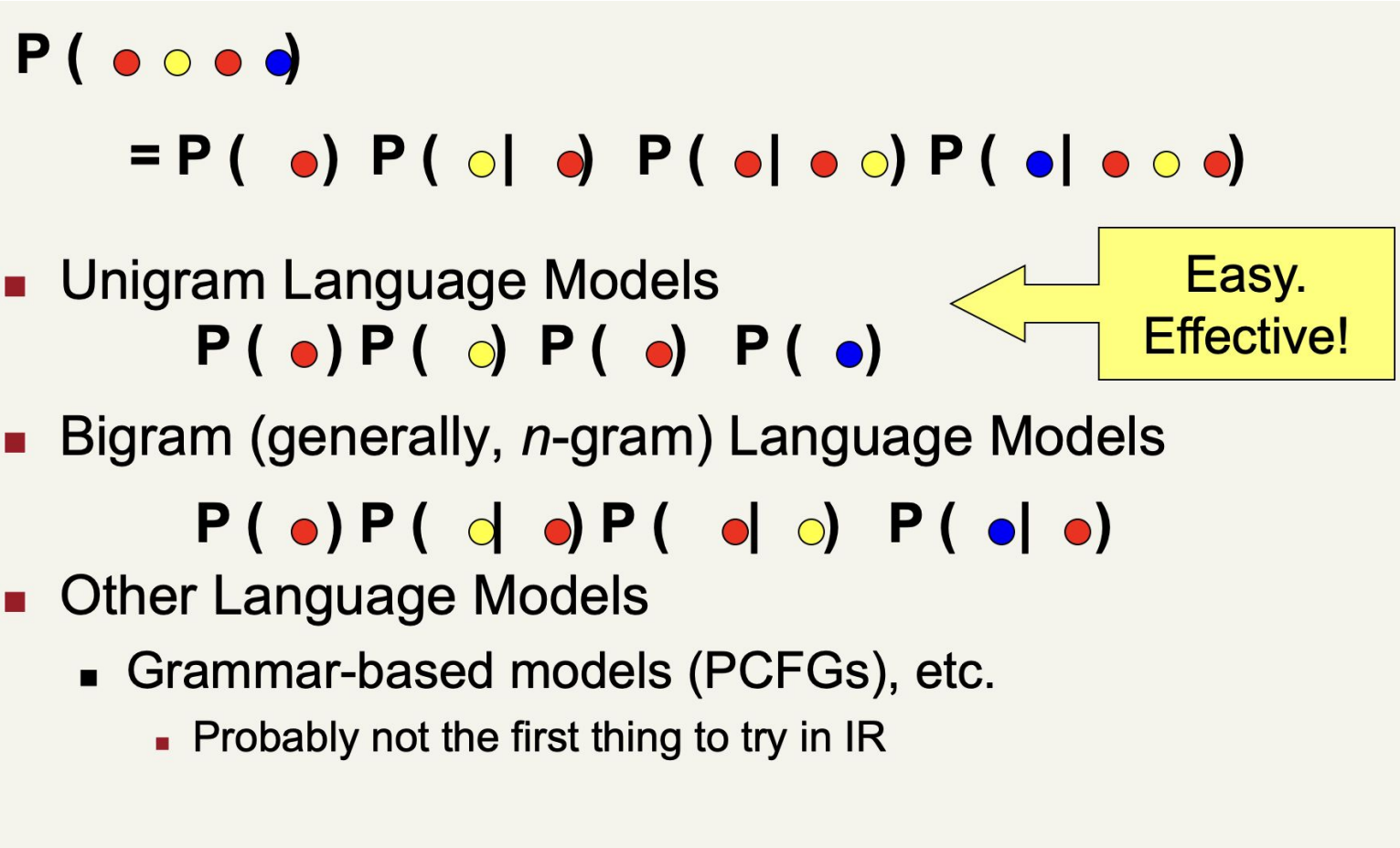
Stochastic Language Models

- A statistical model for generating text
  - Probability distribution over strings in a given language



$$P(\bullet\bullet\bullet\bullet \mid M) = P(\bullet \mid M)\, P(\bullet \mid M, \bullet)\, P(\bullet \mid M, \bullet\bullet)\, P(\bullet \mid M, \bullet\bullet\bullet)$$

Unigram and higher-order models



Generating Shakespeare

U	• 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
Bigram	• 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 first gentleman? •Enter Menenius, if it so many good direction found'st thou art a strong upon command of fear not a liberal largess given away, Falstaff! Exeunt
Trigram	• 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.
Quadrigram	• 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.

GPT-2: Zero-shot Results

DATASET	METRIC	OUR RESULT	PREVIOUS RECORD	HUMAN
Winograd Schema Challenge	accuracy (+)	**70.70%**	63.7%	92%+
LAMBADA	accuracy (+)	**63.24%**	59.23%	95%+
LAMBADA	perplexity (–)	**8.6**	99	~1-2
Children's Book Test Common Nouns (validation accuracy)	accuracy (+)	**93.30%**	85.7%	96%
Children's Book Test Named Entities (validation accuracy)	accuracy (+)	**89.05%**	82.3%	92%
Penn Tree Bank	perplexity (–)	**35.76**	46.54	unknown
WikiText-2	perplexity (–)	**18.34**	39.14	unknown
enwik8	bits per character (–)	**0.93**	0.99	unknown
text8	bits per character (–)	**0.98**	1.08	unknown
WikiText-103	perplexity (–)	**17.48**	18.3	unknown

References

1. Slides provided by Sougata Saha (Instructor, Fall 2022 - CSE 4/535)
2. Materials provided by Dr. Rohini K Srihari
3. https://nlp.stanford.edu/IR-book/information-retrieval-book.html