Introduction to

Informa I

Lecture 9: Probabilistic M guage Model

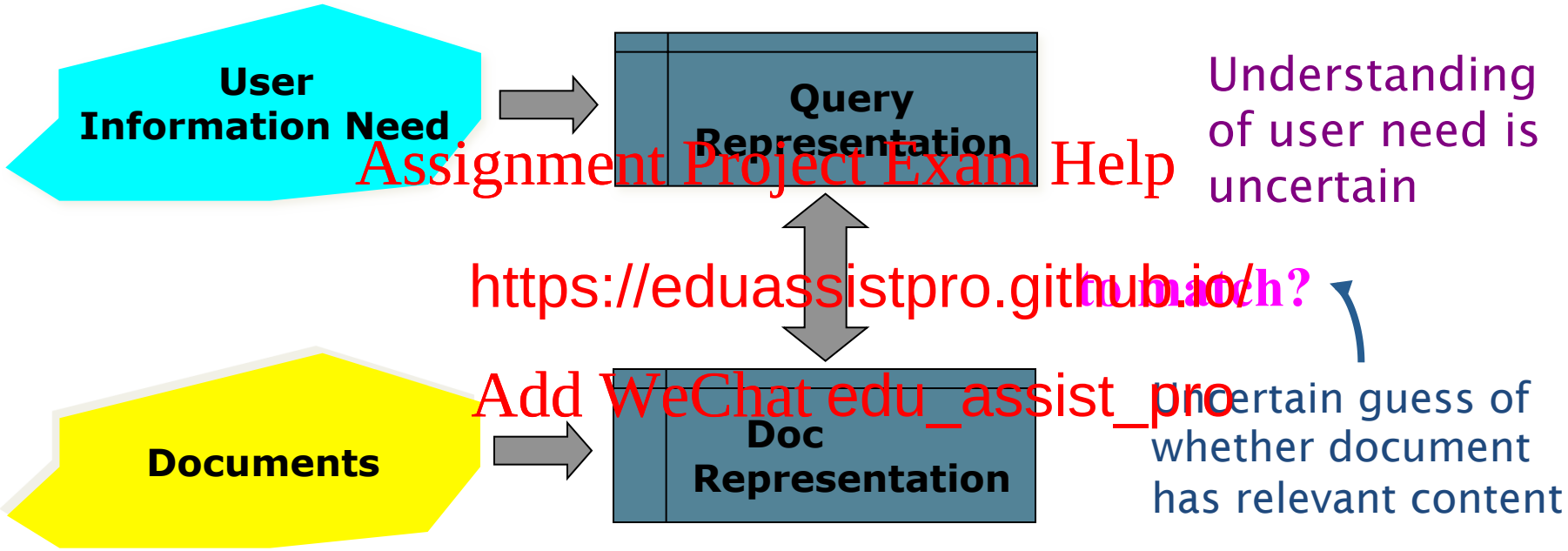
Recap of the last lecture

- Improving search results
  - Especially for high recall. E.g., searching for *aircraft* so it matches with *plane*; *thermodynamic* with *heat*
- Options for i
  - Global meth
    - Query expansion
      - Thesauri
      - Automatic thesaurus generation
    - Global indirect relevance feedback
  - Local methods
    - Relevance feedback
    - Pseudo relevance feedback

Probabilistic relevance feedback

- Rather than reweighting in a vector space…
- If user has told us some relevant and some irrelevant documents, then we can proceed to build a probabilistic c ve Bayes model:
  - $P(t_k|R) = |\mathbf{D}_{rk}| / |\mathbf{D}_r|$
  - $P(t_k|NR) = |\mathbf{D}_{nrk}| / |\mathbf{D}_{nr}|$
    - t_k is a term; $\mathbf{D}_r$ is the set of known relevant documents; $\mathbf{D}_{rk}$ is the subset that contain t_k; $\mathbf{D}_{nr}$ is the set of known irrelevant documents; $\mathbf{D}_{nrk}$ is the subset that contain t_k.

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?

Probabilistic IR topics

- Classical probabilistic retrieval model
  - Probability ranking principle, etc.
- (Naïve) Baye on
- Bayesian ne al
- Language model approac
  - An important emphasis in recent work

- *Probabilistic methods are one of the oldest but also one of the currently hottest topics in IR.*
  - *Traditionally: neat ideas, but they've never won on performance. It may be different now.*

The document ranking problem

- We have a collection of documents
- User issues a query
- A list of docu rned
- **Ranking meth tem.**
  - **In what order do we presen 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|document$_i$, query)

Recall a few probability basics

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

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

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

Prior

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

Posterior

- Odds:

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

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 probabi ately as possible on the basis of whate ilable to the system for this purpose, *t e system to its user will be the best that is obtainable o those data*."

- [1960s/1970s] S. Robertson, W.S. Cooper, M.E. Maron; van Rijsbergen (1979:113); Manning & Schütze (1999:538)

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(document x is **relevant.**

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

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

) - 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.

Probability Ranking Principle (PRP)

- Simple case: no selection costs or other utility concerns that would differentially weight errors

- ***Bayes' Optim***
  - ***x*** is **relevant**

- PRP in action: Rank all docu $p(R|x)$

- Theorem:
  - Using the PRP is optimal, in that it minimizes the loss (Bayes risk) under 1/0 loss
  - Provable if all probabilities correct, etc. [e.g., Ripley 1996]

Probability Ranking Principle

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

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

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**

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 later del
- Questionable
  - "Relevance" of each documen ndent of relevance of other documen
    - Really, it's bad to keep on returning **duplicates**
  - Boolean model of relevance
  - That one has a single step information need
    - Seeing a range of results might let user refine query

$$MMR \stackrel{\text{def}}{=} Arg \max_{D_i \in R \setminus S} \left[\lambda (Sim_1(D_i, Q) - (1-\lambda) \max_{D_j \in S} Sim_2(D_i, D_j)) \right]$$

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 rtson\
- Combine to find document robability
- 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 w other relevant documents.

By making assumptions about tion 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):
  - $\vec{x} = (x_1, \ldots,$
  - $x_i = 1$ iff te
- **"Independence":** terms occur in d ependently
- Different documents can be model ector

- Bernoulli Naive Bayes model (cf. text categorization!)

Binary Independence Model

- Queries: binary term incidence vectors
- Given query ***q***,
  - for each document ***d*** need to compute ***p(R|q,d)***.
  - replace with $\vec{x}$ is binary term incidence ve
- Will use odds and Bayes' R

st nking

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

Binary Independence Model

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

Needs estimation

- Using **Independence** Assum

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

- 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 e

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

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

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

Then...

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 match

Non-matching query terms

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

All matching terms

All query terms

Binary Independence Model

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

O — to be estimated for rankings

- Retrieval Status Value:

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

$$= \sum_{x_i = q_i = 1} \left(\log(odds(p_i)) - \log(odds(r_i)) \right) = \sum_{x_i = q_i = 1} \left(\text{logit}(p_i) - \text{logit}(r_i) \right)$$

Binary Independence Model

- All boils down to computing RSV.

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

$$RSV = \sum_{x_i = q_i = 1} c_i;$$

$$c_i = \log \frac{\text{odds}(p_i)}{\text{odds}(r_i)} = \log \text{it}(p_i) - \log \text{it}(r_i)$$

So, how do we compute c_i's from our data ?

Binary Independence Model

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

Documents	Relevant	Non-Relevant	Total
$X_i = 1$			n
$X_i = 0$			N-n
Total	S		

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

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

However, these estimates could be 0.

Add ½ Smoothing

- Add ½ to each of the center four cells.

Documents	Relevant	Non-Relevant	Total
$X_i = 1$			n+1
$X_i = 0$			N-n+1
Total	S+1		+2

$$c_i \approx K(N,n,S,s) = \log \frac{(s+1/2)/(S-s+1/2)}{(n-s+1/2)/(N-n-S+s+1/2)}$$

Example /1

- Query = $\{x_1, x_2\}$
- $O(R=1|D_3, q)$

Doc	Judgment	x_1	x_2	x_3
D_1	R	1	1	1
D_2	R	0	0	1
D_3	R	1	0	0
D_4	NR	1	0	1
D_5	NR	0	1	1

Example /2

- Estimate p_i and r_i

Doc	Judgment	x_1	x_2	x_3
D_1	R	1	1	1
D_2	R	0	0	1
D_3	R	1	0	0
D_4	NR	1	0	1
D_5	NR	0	1	1

in fact, tf-idf can be deemed as the cross-entropy

Estimation – key challenge

- If non-relevant documents are approximated by the whole collection, then r_i (prob. of occurrence in non-relevant documents for query) *is n/N* and
  - log $(1-r_i)/r$ IDF!
- p_i (probabili vant documents) can be estimated in various ways:
  - from relevant documents if know some
    - Relevance weighting can be used in feedback loop
  - constant (Croft and Harper combination match – 0.5) – then just get idf weighting of terms
  - proportional to prob. of occurrence in collection
    - more accurately, to log of this (Greiff, SIGIR 1998)

optional

Iteratively estimating p_i

1. Assume that p_i constant over all x_i in query
   - p_i = 0.5 (even odds) for any given doc
2. Determine guess of relevant document set:
   - V is fixed si ocuments on this model (not
3. We need to improve our g p_i and r_i, so
   - Use distribution of x_i in docs in V. Let V_i be set of documents containing x_i
     - $p_i = |V_i| / |V|$
   - Assume if not retrieved then not relevant
     - $r_i = (n_i - |V_i|) / (N - |V|)$
4. Go to 2. until converges then return ranking

optional

Probabilistic Relevance Feedback

1. Guess a preliminary probabilistic description of *R* and use it to retrieve a first set of documents V, as above.
2. Interact with description: learn some d and NR
3. Reestimate p_i and r_i on th hese
   - Or can combine new information with original guess (use Bayesian prior):

$$p_i^{(2)} = \frac{|V_i| + \kappa p_i^{(1)}}{|V| + \kappa}$$

κ is prior weight

4. Repeat, thus generating a succession of approximations to *R*.

PRP and BIR

- Getting reasonable approximations of probabilities is possible.
- Requires r :
  - ***term ind***
  - ***terms not in query don't utcome***
  - ***boolean representation ts/queries/ relevance***
  - ***document relevance values are independent***
- Some of these assumptions can be removed
- Problem: either require partial relevance information or only can derive somewhat inferior term weights

Okapi BM25

- Heuristically extend the BIR to include information of term frequencies, document length, etc.

contribution of *tf*

$$RSVd = \sum_{t \in q} \left(\log \frac{}{df_t} \right) \frac{}{k_1 \left((1-b) \frac{}{L_{ave}} \right) tf_{t,d}} \cdot \frac{(k_3+1)tf_{t,q}}{k_3 + tf_{t,q}}$$

idf

Normalized term freq (doc)

Normalized term freq (query)

- Typically, $k_1, k_3 \in [1.2, 2.0], b = 0.75$

Good and Bad News

- Standard Vector Space Model
  - Empirical for the most part; success measured by results
  - Few properties provable
- Probabilistic Mo
  - Based on a firm
  - Theoretically justified optimal rank
- Disadvantages
  - Making the initial guess to get V
  - Binary word-in-doc weights (not using term frequencies)
  - Independence of terms (can be alleviated)
  - Amount of computation
  - Has never worked convincingly better in practice

Resources

S. E. Robertson and K. Spärck Jones. 1976. Relevance Weighting of Search Terms. *Journal of the American Society for Information Sciences* 27(3): 129–146.

C. J. van Rijsbergen. 1979. *Information Retrieval*. 2nd ed. London: Butterworths, www.dcs.gla.a h] http://

N. Fuhr. 1992. Pro n Retrieval. *The Computer Journal*, 35(3),243-255. [Easiest r

F. Crestani, M. Lalmas, C. J. van Rijsbergen, and I. Campbell. 1998. Is This Document Relevant? ... Probably: A Survey of Probabilistic Models in Information Retrieval. *ACM Computing Surveys* 30(4): 528–552.

http://www.acm.org/pubs/citations/journals/surveys/1998-30-4/p528-crestani/

[Adds very little material that isn't in van Rijsbergen or Fuhr]

Resources

H.R. Turtle and W.B. Croft. 1990. Inference Networks for Document Retrieval. *Proc. ACM SIGIR*: 1-24.

E. Charniak. Bayesian nets without tears. *AI Magazine* 12(4): 50-63 (1991). *http://www.aaai.org/Libra l*

D. Heckerman. 1995 sian Networks. Microsoft Technical Report MSR-TR-95-06 *http://www.research.microsoft.com/~hecker*

N. Fuhr. 2000. Probabilistic Datalog: Implementing Logical Information Retrieval for Advanced Applications. *Journal of the American Society for Information Science* 51(2): 95–110.

R. K. Belew. 2001. *Finding Out About: A Cognitive Perspective on Search Engine Technology and the WWW*. Cambridge UP 2001.

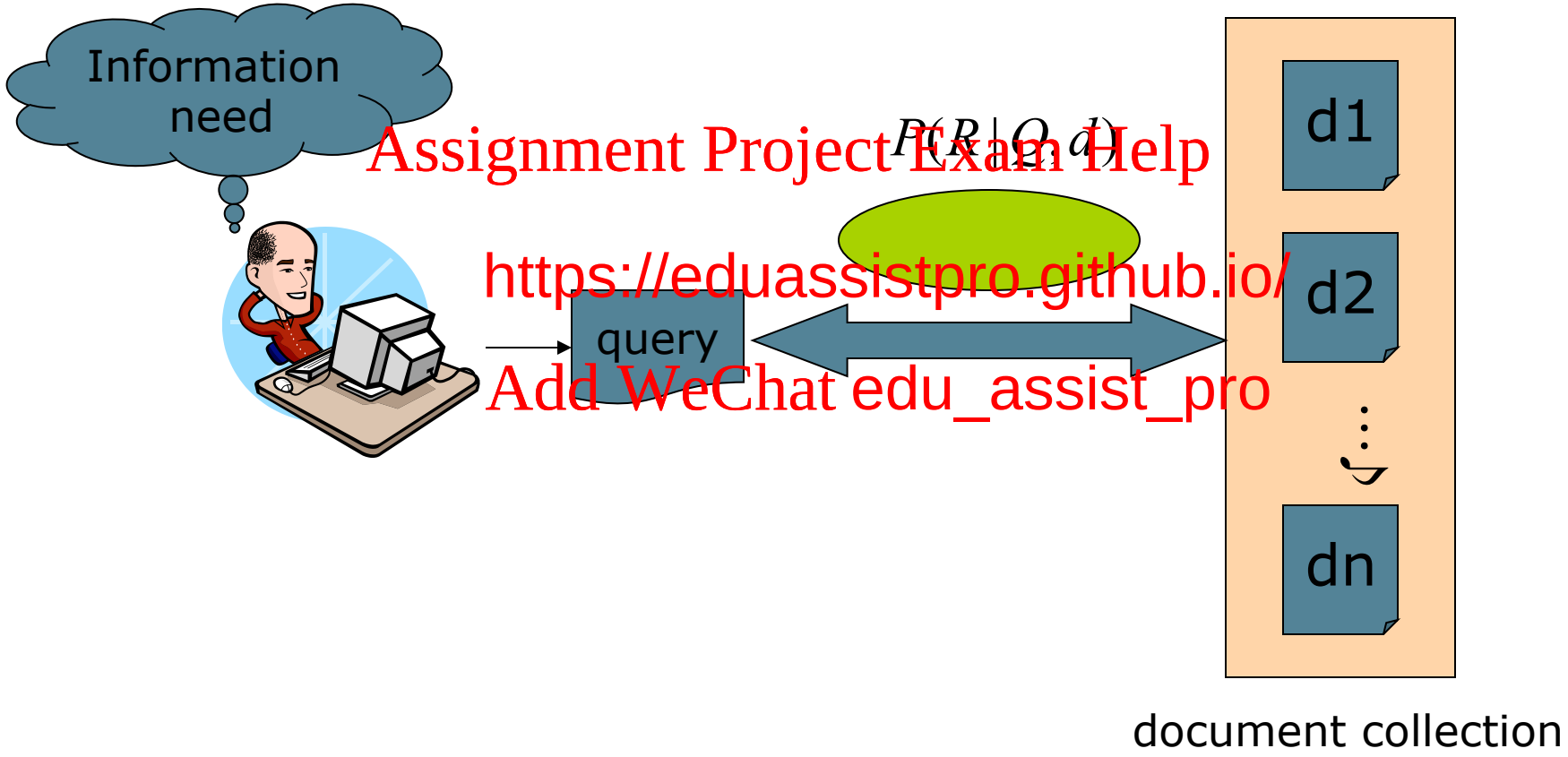
MIR 2.5.4, 2.8

LANGUAGE MODEL

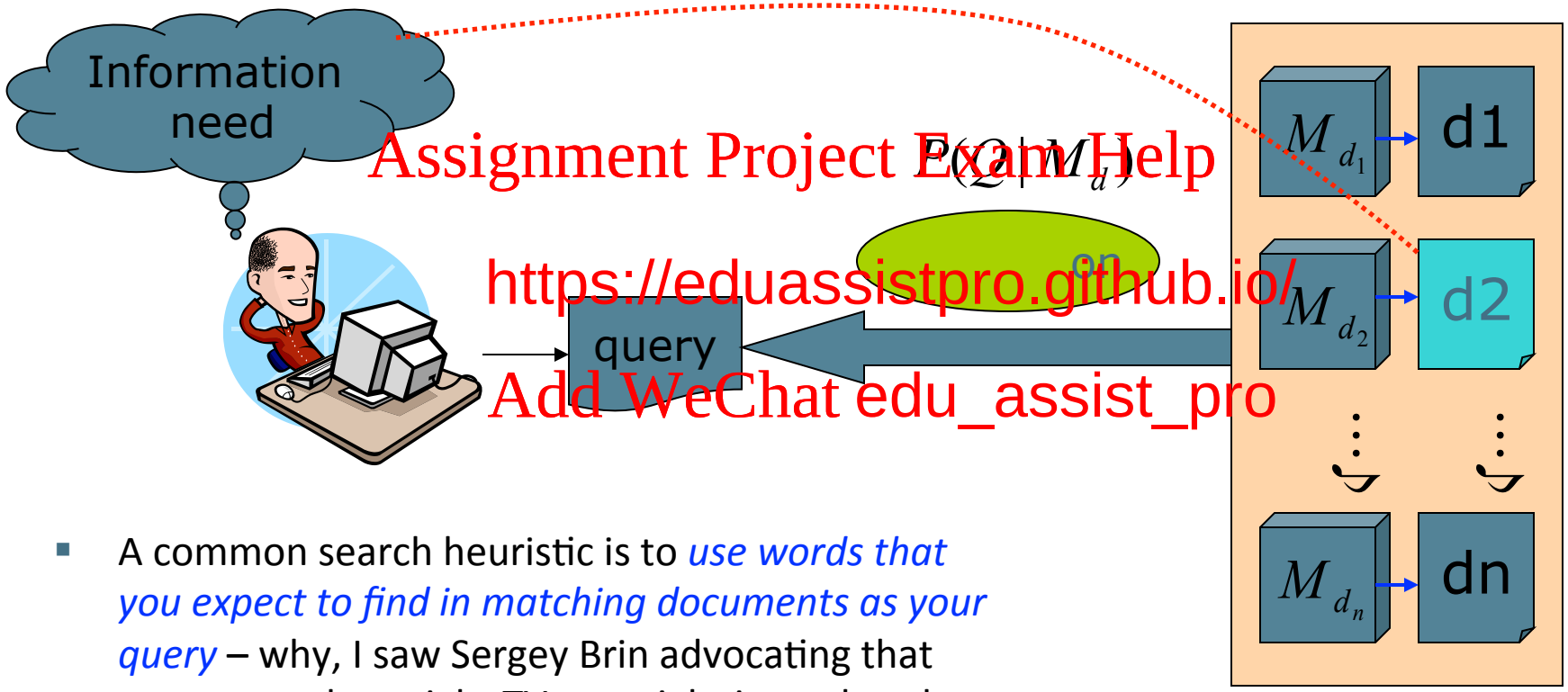
Today

- The Language Model Approach to IR
  - Basic query generation model
  - Alternative models

Standard Probabilistic IR



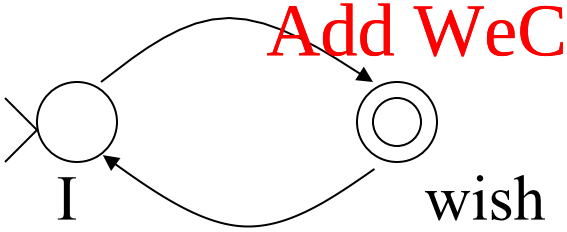
IR based on Language Model (LM)



- 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!
- See later slides for a more formal justification

Formal Language (Model)

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



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

s: the man likes the woman

0.2 0.01 0.02 0.2 0.01

multiply

$P(s \mid M) = 0.00000008$

Stochastic Language Models

- Model *probability* of generating any string

Model M1	
0.2	the
0.01	class
0.0001	sayst
0.0001	pleaseth
0.0001	yon
0.0005	maiden
0.01	woman

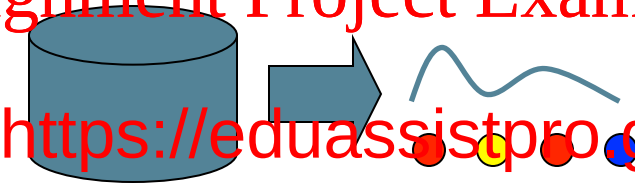
0.2	the
0.0001	class
0.03	sayst
0.02	pleaseth
0.1	yon
0.01	maiden
0.0001	woman

t	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 (● ○ ● ● | M) = P (● P (○ | M, ●)

P (● | M, ● ○)

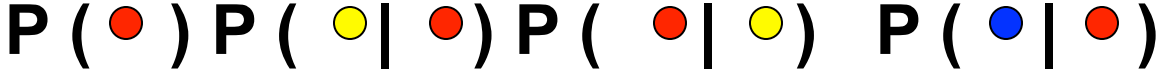
P (● | M, ● ○ ●)

Unigram and higher-order models

Easy. Effective!

- Unigram Lang

- Bigram (generally, *n*-gram) Language Models



- Other Language Models
  - Grammar-based models (PCFGs), etc.
    - Probably not the first thing to try in IR

Using Language Models in IR

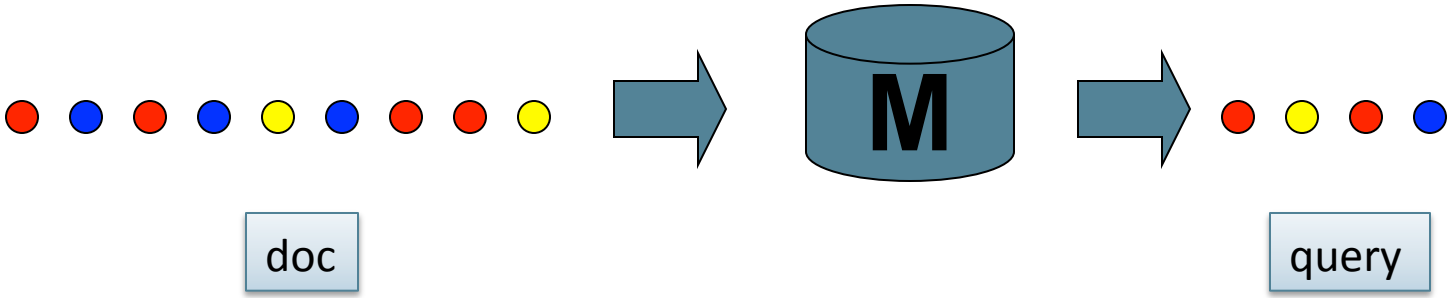
- Treat each document as the basis for a model (e.g., unigram sufficient statistics)
- Rank document d based on P(d | q)
- P(d | q) = P(q
  - P(q) is the same for all docu ore
  - P(d) [the prior] is often treat me for all d
    - But we could use criteria like authority, length, genre
  - P(q | d) is the probability of q given d's model
- Very general formal approach

The fundamental problem of LMs

- Usually we don't know the model **M**
  - But have a sample of text representative of that model

P ( )

- Estimate a language model ple
- Then compute the observation probability



Language Models for IR

- Language Modeling Approaches
  - Attempt to **model query generation process**
  - Documents are ranked by **the probability that a query would be ob le from the respective d**
    - Multinomial approach

$$P(Q|M_D) = \prod_w P(w|M_D)^{q_w}$$

Retrieval based on probabilistic LM

- Treat the generation of queries as a random process.
- Approach
  - Infer a language model for each document.
  - Estimate the the query according to each of th
  - Rank the documents accordi nrobabilities.
  - Usually a unigram estimate o sed
    - Some work on bigrams, paralleling van Rijsbergen

Retrieval based on probabilistic LM

- Intuition
  - Users …
    - Have a reasonable idea of terms that are likely to occur in documents of interest.
    - They will ch ish these documents from others
- Collection statistics …
  - Are integral parts of the language model.
  - Are not used heuristically as in many other approaches.
    - In theory. In practice, there's usually some wiggle room for empirically set parameters

Query generation probability (1)

- Ranking formula

$$p(Q,d) = p(d)\,p(Q \mid d)$$

$$\approx p(d)\,p(Q \mid M_d)$$

- The probability of the language model of document d using

$$\hat{p}(Q \mid M_d) = \prod_{t \in Q} \hat{p}_{ml}(t \mid M_d)$$

$$= \prod_{t \in Q} \frac{tf_{(t,d)}}{dl_d}$$

Unigram assumption:
Given a particular language model, the query terms occur independently

M_d : language model of document d

$tf_{(t,d)}$: raw tf of term t in document d

dl_d : total number of tokens in document d

Insufficient data

- Zero probability $p(t \mid M_d) = 0$
  - May not wish to assign a probability of zero to a document that is missing one or more of the query terms [gives conjunction sema
- General appr
  - A non-occurring term is possible but more likely than would be expected by chanc ction.
  - If $tf_{(t,d)} = 0$, $p(t \mid M_d) = \frac{cf_t}{cs}$

cf_t : raw count of term t in the collection

cs : raw collection size(total number of tokens in the collection)

Insufficient data

- Zero probabilities spell disaster
  - We need to smooth probabilities
    - Discount nonzero probabilities
    - Give some gs
- There's a wid to smoothing probability distributions to his problem, such as adding 1, ½ or ε to richlet priors, discounting, and interpolation
  - [See FSNLP ch. 6 or CS224N if you want more]
- A simple idea that works well in practice is to use a mixture between the document multinomial and the collection multinomial distribution

Mixture model

- Jelinek-Mercer method
  - $P(w|d) = \lambda P_{mle}(w|M_d) + (1 - \lambda)P_{mle}(w|M_c)$
- Mixes the probability from the document with the general collec word
- Correctly setting λ is very i
- A high value of lambda mak rch "conjunctive-like" – suitable for short queries
- A low value is more suitable for long queries
- Can tune λ to optimize performance
  - Perhaps make it dependent on document size (cf. Dirichlet prior or Witten-Bell smoothing)

Basic mixture model summary

- General formulation of the LM for IR

$$p(Q,d) = p(d)\prod_{t\in Q}((1-\lambda)p(t) + \lambda p(t|M_d))$$

collection/backgroun

individual- el

- The user has a document in mind, and generates the query from this document.
- The equation represents the probability that the document that the user had in mind was in fact this one.

Relationship to idf

Note here (i.e., [CMS09]) λ is multiplied to the background model.

$f_{qi,D}$=0 ➔ query word that does not occur in the doc

Add contributions from i:$f_{qi,D}$>0

Becomes a constant | Q, C

proportional to the tf, inversely proportional to the cf

Example

- Document collection (2 documents)
  - d_1: Xerox reports a profit but revenue is down
  - d_2: Lucent narrows quarter loss but revenue decreases further
- Model: MLE u ts; λ = ½
- Query: *revenue down*
  - $P(Q|d_1)$ = [(1/8 + 2/16)/2] x [(1/8 + 1/16)/2]
    = 1/8 x 3/32 = 3/256
  - $P(Q|d_2)$ = [(1/8 + 2/16)/2] x [(0 + 1/16)/2]
    = 1/8 x 1/32 = 1/256
- Ranking: $d_1 > d_2$

Ponte and Croft Experiments

- Data
  - TREC topics 202-250 on TREC disks 2 and 3
    - Natural language queries consisting of one sentence each
  - TREC topics 51-100 on TREC disk 3 using the concept fields
    - Lists of good

<num>
<dom> ational Economics
<title>Topic: Satellite Launch Contracts
<desc>Description:
… </desc>

<con>Concept(s):
1. Contract, agreement
2. Launch vehicle, rocket, payload, satellite
3. Launch services, … </con>

Precision/recall results 202-250

Precision/recall results 51-100♪

Language models: pro & con

- Novel way of looking at the problem of text retrieval based on probabilistic language modeling
  - Conceptually simple and explanatory
  - Formal mat
  - Natural use ristics (almost…)
- LMs provide be improved to the extent that the following con be met
  - Our language models are accurate representations of the data.
  - Users have some sense of term distribution.*
    - *Or we get more sophisticated with translation model

Comparison With Vector Space

- There's some relation to traditional tf.idf models:
  - (unscaled) term frequency is directly in model
  - the probabilities do length normalization of term frequencies
  - the effect of all collection frequencies is a little like idf: in the general collection but common in so ts will have a greater influence on the ranking

Comparison With Vector Space

- Similar in some ways
  - Term weights based on frequency
  - Terms often used as if they were independent
  - Inverse docu y used
  - Some form o ful
- Different in others
  - Based on probability rather than similarity
    - Intuitions are probabilistic rather than geometric
  - Details of use of document length and term, document, and collection frequency differ

Resources

J.M. Ponte and W.B. Croft. 1998. A language modelling approach to information retrieval. In *SIGIR 21.*

D. Hiemstra. 1998. A linguistically motivated probabilistic model of information retrieval. *ECDL 2*, pp. 569–584.

A. Berger and J. Lafferty. stical translation. *SIGIR 22*, pp. 222–229.

D.R.H. Miller, T. Leek, an arkov model information retrieval system. *SIGIR 22*, pp. 214–221.

[Several relevant newer papers at *SIGIR 23–25*

Workshop on Language Modeling and Information Retrieval, CMU 2001. http://la.lti.cs.cmu.edu/callan/Workshops/lmir01/ .

The Lemur Toolkit for Language Modeling and Information Retrieval. http://www-2.cs.cmu.edu/~lemur/ . CMU/Umass LM and IR system in C(++), currently actively developed.