CAB431
Search Engine Technology

Information Retrieval models

Queensland University of Technology

Outline

1. *Overview of IR Models*
2. *Older IR Models*
   - *Boolean Retrieval*
   - *Vector Space Model*
3. Probabilistic Models
   - BM25
   - Language models
4. *Relevance models*
   - *Pseudo-Relevance Feedback*
   - *KL-Divergence*

1. Overview of IR Models

- Information retrieval (IR) models Provide a mathematical framework for defining the search process
  - includes explanation of assumptions
  - basis of many ranking algorithms
  - can be implicit
- For a given query *Q*, an IR model finds relevant documents to answer *Q*.
- Progress in IR models has corresponded with improvements in effectiveness.
- Theories about relevance

Relevance

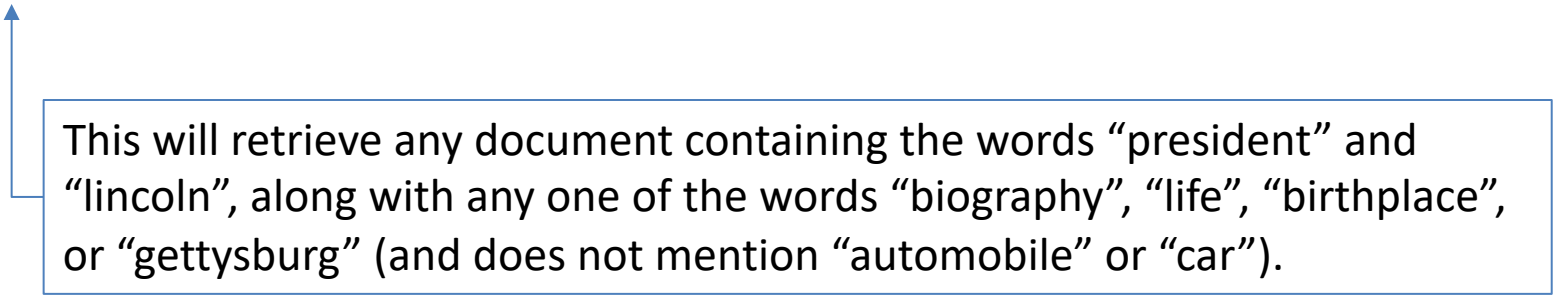
- Complex concept that has been studied for some time
  - Many factors to consider
  - People often disagree when making ***relevance judgments***
- IR models make various assumptions about relevance to simplify problem
  - e.g., *topical* vs. *user* relevance
    - A document is topically relevant to a query if it is judged to be on the same topic, i.e., the query and the document are about the same thing.
    - User relevance considers all the other factors that go into a user's judgment of relevance.
  - e.g., *binary* vs. *multi-valued* relevance

2. Order IR Models

- Boolean Retrieval
  - Two possible outcomes for query processing
    - TRUE and FALSE
    - “exact-match” retrieval
    - simplest form of ranking
  - Query usually specified using Boolean operators
    - AND, OR, NOT,
    - Proximity operators (define constraints on the distance between words; e.g., requiring words to co-occur within a certain “window” (length) of text); and
    - Wildcard characters (define the minimum string match required for a word) are also commonly used in Boolean queries.

Searching by Numbers

- It is the process of developing queries with a focus on the size of the retrieved set.
- It is a consequence of the limitations of the Boolean retrieval model.
- For example, sequence of queries driven by number of retrieved documents
  - e.g. “lincoln” search of news articles
  - president AND lincoln
  - president AND lincoln AND NOT (automobile OR car)
  - president AND lincoln AND biography AND life AND birthplace AND gettysburg AND NOT (automobile OR car)
  - president AND lincoln AND (biography OR life OR birthplace OR gettysburg) AND NOT (automobile OR car)



Boolean Retrieval cont.

- Advantages
  - Results are predictable, relatively easy to explain to users
  - Many different features can be incorporated (e.g., document date or type)
  - Efficient processing since many documents can be eliminated from search
- Disadvantages
  - Effectiveness depends entirely on user (simple queries will not work well).
  - Complex queries are difficult that requires considerable experience.

Vector Space Model

- Documents and query represented by a vector of term weights

$$Q = (q_1, q_2, \ldots, q_t)$$

$$D_i = (d_{i1}, d_{i2}, \ldots, d_{it})$$

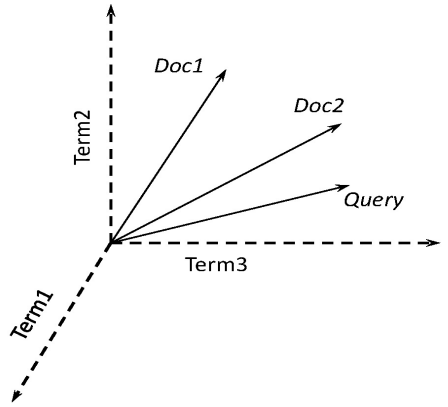
- A Collection of Documents can be represented by a matrix of **term weights**

	$Term_1$	$Term_2$	$\ldots$	$Term_t$
Doc_1	d_{11}	d_{12}	$\ldots$	d_{1t}
Doc_2	d_{21}	d_{22}	$\ldots$	d_{2t}
$\vdots$	$\vdots$			
Doc_n	d_{n1}	d_{n2}	$\ldots$	d_{nt}

- Example (in inverted indexing)
- 3-d pictures useful, but can be misleading for high-dimensional space

D_1 Tropical Freshwater Aquarium Fish.
D_2 Tropical Fish, Aquarium Care, Tank Setup.
D_3 Keeping Tropical Fish and Goldfish in Aquariums, and Fish Bowls.
D_4 The Tropical Tank Homepage - Tropical Fish and Aquariums.

Terms	Documents			
	D_1	D_2	D_3	D_4
aquarium	1	1	1	1
bowl	0	0	1	0
care	0	1	0	0
fish	1	1	2	1
freshwater	1	0	0	0
goldfish	0	0	1	0
homepage	0	0	0	1
keep	0	0	1	0
setup	0	1	0	0
tank	0	1	0	1
tropical	1	1	1	2



Vector Space Model - similarity measure

- Documents ranked by distance between points representing query and documents
  - *Similarity* measure more common than a distance or *dissimilarity* measure
  - e.g., Cosine correlation

$$Cosine(D_i, Q) = \frac{\sum_{j=1}^{t} d_{ij} \cdot q_j}{\sqrt{\sum_{j=1}^{t} {d_{ij}}^2 \cdot \sum_{j=1}^{t} {q_j}^2}}$$

$$Q = (q_1, q_2, \ldots, q_t) \qquad D_i = (d_{i1}, d_{i2}, \ldots, d_{it})$$

Example of Similarity Calculation

- Consider two documents D_1, D_2 and a query Q
  - D_1 = (0.5, 0.8, 0.3), D_2 = (0.9, 0.4, 0.2), Q = (1.5, 1.0, 0)

$$Cosine(D_1, Q) = \frac{(0.5 \times 1.5) + (0.8 \times 1.0)}{\sqrt{(0.5^2 + 0.8^2 + 0.3^2)(1.5^2 + 1.0^2)}}$$
$$= \frac{1.55}{\sqrt{(0.98 \times 3.25)}} = 0.87$$

$$Cosine(D_2, Q) = \frac{(0.9 \times 1.5) + (0.4 \times 1.0)}{\sqrt{(0.9^2 + 0.4^2 + 0.2^2)(1.5^2 + 1.0^2)}}$$
$$= \frac{1.75}{\sqrt{(1.01 \times 3.25)}} = 0.97$$

Document Representation - *Term Weights*

- *Tf*idf* weight
  - Term frequency weight measures importance of term k in document (D_i): $tf_{ik} = \frac{f_{ik}}{\sum_{j=1}^{t} f_{ij}}$
  - Inverse document frequency measures importance in collection: $idf_k = \log \frac{N}{n_k}$
  - Some heuristic modifications

$$d_{ik} = \frac{(\log(f_{ik}) + 1) \cdot \log(N/n_k)}{\sqrt{\sum_{x=1}^{t}[(\log(f_{ix}) + 1) \cdot \log(N/n_x)]^2}}$$

$$d_{ik} = \frac{(\log(f_{ik})+1)\cdot\log(N/n_k)}{\sqrt{\sum_{k=1}^{t}[(\log(f_{ik})+1.0)\cdot\log(N/n_k)]^2}}$$

Incorrect Eq in the text-book

Query Representation - *Rocchio algorithm*

- Relevance Feedback, a technique for query modification based on user-identified relevant documents.
- Rocchio algorithm (based on *Optimal query*)
  - Maximizes the difference between the average vector representing the relevant documents and the average vector representing the non-relevant documents.
- Modifies query *Q* to *Q'* according to

$$q'_j = \alpha.q_j + \beta.\frac{1}{|Rel|}\sum_{D_i \in Rel} d_{ij} - \gamma.\frac{1}{|Nonrel|}\sum_{D_i \in Nonrel} d_{ij}$$

  - α, β, and γ are parameters
    - Typical values 8, 16, 4
  - q_j is the initial weight of query term j,
  - d_{ij} is the weight of the jth term in document i,
  - *Rel* is the set of identified relevant documents,
  - *Nonrel* is the set of non-relevant documents,
  - | . | gives the size of a set.

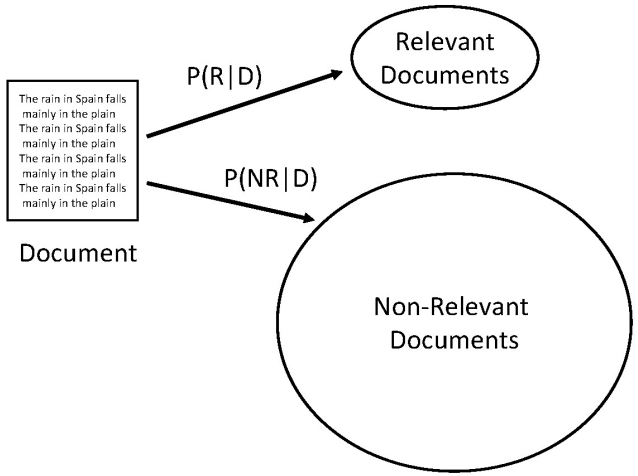
Vector Space Model cont.

- Advantages
  - Simple computational framework for ranking
  - Any similarity measure or term weighting scheme could be used
- Disadvantages
  - Assumption of term independence
  - No *predictions* about techniques for effective ranking. There is an implicit assumption that relevance is related to the similarity of query and document vectors.

3. Probabilistic Models

- It is hard to prove that an IR model will achieve better effectiveness than any other model since we are trying to formalize a complex human activity.
- The validity of a retrieval model generally has to be validated empirically, rather than theoretically.
- **Probability Ranking Principle** (early theoretical statement about effectiveness, Robertson (1977)). "Originally described as follow":
  - "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."

IR as Classification - Bayes Classifier



- $P(R|D)$ is a conditional probability representing the probability of relevance for the given document D, and
- $P(NR|D)$ is the conditional probability of non-relevance.

- Bayes Decision Rule
  - A document *D* is relevant if $P(R|D) > P(NR|D)$
- Estimating probabilities
  - use Bayes Rule

$$P(R|D) = \frac{P(D|R)P(R)}{P(D)}$$

    - P(R) is the priori probability of relevance
  - classify a document as relevant if

$$\frac{P(D|R)}{P(D|NR)} > \frac{P(NR)}{P(R)}$$

    - The left-hand side (lhs) is *likelihood ratio.*

Examples of Relevant and Non-relevant documents

- Let Q = {US, ECONOM, ESPIONAG} be a query
- $\boldsymbol{C}$ = {D_1, D_2, D_3, D_4, D_5, D_6, D_7} be a collection of documents, where

  D_1 = {GERMAN, VW}

  D_2 = {US, US, ECONOM, SPY}

  D_3 = {US, BILL, ECONOM, ESPIONAG}

  D_4 = {US, ECONOM, ESPIONAG, BILL}

  D_5 = {GERMAN, MAN, VW, ESPIONAG}

  D_6 = {GERMAN, GERMAN, MAN, VW, SPY}

  D_7 = {US, MAN, VW}
- $P(R|D_5)$, $P(R|D)$? Where D = {US, VW, ESPIONAG} = {d_1, d_2, d_3}
- $P(D_5|R)$, $P(D|R)$?

Document ID	Terms: d_{ij}	Relevance to Q
D_1	GERMAN, VW	0 no
D_2	US, US, ECONOM, SPY	1 yes
D_3	US, BILL, ECONOM, ESPIONAG	1 yes
D_4	US, ECONOM, ESPIONAG, BILL	1 yes
D_5	GERMAN, MAN, VW, ESPIONAG	0 no
D_6	GERMAN, GERMAN, MAN, VW, SPY	0 no
D_7	US, MAN, VW	0 no

Estimating P(D|R)

- Assume independence

$$P(D|R) = \prod_{i=1}^{t} P(d_i|R)$$

- *Binary independence model*
  - document represented by a vector of binary features indicating term occurrence (or non-occurrence). E.g.,
    - *T* = {US, BILL, ECONOM, ESPIONAG, GERMAN, MAN, VW, SPY }
    - *D* = {US, VW, ESPIONAG}
    - The binary vector of *D* = (1, 0, 0, 1, 0, 0, 1, 0)
  - p_i is probability that term *i* occurs (i.e., has value 1) in relevant document, s_i is probability of occurrence in non-relevant document

Binary Independence Model

$$\frac{P(D|R)}{P(D|NR)} = \prod_{i:d_i=1} \frac{p_i}{s_i} \cdot \prod_{i:d_i=0} \frac{1-p_i}{1-s_i}$$

$$= \prod_{i:d_i=1} \frac{p_i}{s_i} \cdot \left(\prod_{i:d_i=1} \frac{1-s_i}{1-p_i} \cdot \prod_{i:d_i=1} \frac{1-p_i}{1-s_i}\right) \cdot \prod_{i:d_i=0} \frac{1-p_i}{1-s_i}$$

$$= \prod_{i:d_i=1} \frac{p_i(1-s_i)}{s_i(1-p_i)} \cdot \prod_{i} \frac{1-p_i}{1-s_i}$$

Where $i{:}d_i = 1$ means terms that have the value 1 (presence) in the document; $i{:}d_i = 0$ means terms that have the value 0 (absence) in the document.

Binary Independence Model

- Scoring function is (if we ignore the common 2nd product)

$$\sum_{i:d_i=1} \log \frac{p_i(1-s_i)}{s_i(1-p_i)}$$

- Query provides information about relevant documents
- If we assume p_i constant, s_i approximated by entire collection, get *idf*-like weight

$$\log \frac{0.5(1-\frac{n_i}{N})}{\frac{n_i}{N}(1-0.5)} = \log \frac{N-n_i}{n_i}$$

Contingency Table

	Relevant	Non-relevant	Total
$d_i = 1$	r_i	$n_i - r_i$	n_i
$d_i = 0$	$R - r_i$	$N - n_i - R + r_i$	$N - n_i$
Total	R	$N - R$	N

r_i is the number of relevant documents containing term *i*
n_i is the number of documents containing term *i*
N is the total number of documents in the collection
R is the number of relevant documents for this query
d_i = 1 if term *i* is present in the document, and 0 otherwise

Example: *term* 1 = 'US'

	Relevant	Non-relevant	Total
$d_1 = 1$	r_i = 3	$n_i - r_i$ = 1	n_i = 4
$d_1 = 0$	$R - r_i$ = 0	$(N-R)-(n_i - r_i) = N - n_i - R + r_i$ = 3	$N - n_i$ = 3
Total	R = 3	N-R = 4	N = 7

$$p_i = (r_i + 0.5)/(R + 1) \qquad s_i = (n_i - r_i + 0.5)/(N - R + 1)$$

Putting these estimates into the scoring function:

$$\sum_{i:d_i=q_i=1} \log \frac{(r_i+0.5)/(R-r_i+0.5)}{(n_i-r_i+0.5)/(N-n_i-R+r_i+0.5)}$$

BM25

- Popular and effective ranking algorithm based on binary independence model
  - adds document and query term weights

$$\sum_{i \in Q} \log \frac{(r_i + 0.5)/(R - r_i + 0.5)}{(n_i - r_i + 0.5)/(N - n_i - R + r_i + 0.5)} \cdot \frac{(k_1+1) f_i}{K + f_i} \cdot \frac{(k_2+1) qf_i}{k_2 + qf_i}$$

  - f_i is the frequency of term i in the document;
  - qf_i is the frequency of term i in the query;
  - *k1, k2* and *K* are parameters whose values are set empirically
  - $K = k_1((1-b) + b \cdot \frac{dl}{avdl})$, *dl* is doc length and *avdl* is the average length of a document in the collection.
  - Typical TREC value for *k1* is 1.2, *k2* varies from 0 to 1000, b = 0.75

BM25 Example

- Query with two terms, “president lincoln”, (*qf = 1)*
- No relevance information (*r and R are* zero)
- N = 500,000 documents
- *“president”* occurs in 40,000 documents (n_1 = 40, 000)
- *“lincoln”* occurs in 300 documents (n_2 = 300)
- “president” occurs 15 times in doc (f_1 = 15)
- *“lincoln”* occurs 25 times (f_2 = 25)
- document length is 90% of the average length (*dl/avdl* = .9)
- k_1 = 1.2, b = 0.75, and k_2 = 100
- $K = 1.2 \cdot (0.25 + 0.75 \cdot 0.9) = 1.11$

BM25 Example

$$
\begin{aligned}
BM25(Q,D) \quad = \quad & \log \frac{(0+0.5)/(0-0+0.5)}{(40000-0+0.5)/(500000-40000-0+0+0.5)} \\
& \times \frac{(1.2+1)15}{1.11+15} \times \frac{(100+1)1}{100+1} \\
& + \log \frac{(0+0.5)/(0-0+0.5)}{(300-0+0.5)/(500000-300-0+0+0.5)} \\
& \times \frac{(1.2+1)25}{1.11+25} \times \frac{(100+1)1}{100+1}
\end{aligned}
$$

$$
\begin{aligned}
= \quad & \log 460000.5/40000.5 \cdot 33/16.11 \cdot 101/101 \\
& + \log 499700.5/300.5 \cdot 55/26.11 \cdot 101/101 \\
= \quad & 2.44 \cdot 2.05 \cdot 1 + 7.42 \cdot 2.11 \cdot 1 \\
= \quad & 5.00 + 15.66 = 20.66
\end{aligned}
$$

BM25 Example

- Effect of term frequencies

Frequency of "president"	Frequency of "lincoln"	BM25 score
15	25	20.66
15	1	12.74
15	0	5.00
1	25	18.2
0	25	15.66

Language Model

- *Unigram language model (a simple language model)*
  - probability distribution over the words in a language.
  - For example, if the documents in a collection contained just five different words (w_1, w_2, ..., w_5), a possible language model for that collection might be
    - (0.2, 0.1, 0.35, 0.25, 0.1)
    - where each number is the probability of a word occurring.
- N-gram language model
  - predicts words based on longer sequences are used. An *n*-gram model predicts a word based on the previous ***n* – 1 words.**
  - The most common n-gram models are bigram (predicting based on the previous word) and trigram (predicting based on the previous two words) models.

Language Model

- We can use language models to represent the topical content of a document.
- A topic is defined as a probability distribution over words – a language model.
- A language model can be used to "generate" new text by sample words according to the probability distribution.
- A *topic* in a document or query can be represented as a language model
  - i.e., words that tend to occur often when discussing a topic will have high probabilities in the corresponding language model

LMs for Retrieval

- 3 possibilities:
  - probability of generating the query text from a document language model
  - probability of generating the document text from a query language model
  - comparing the language models representing the query and document topics
- Models of topical relevance
  - The probability of query generation is the measure of how likely it is that a document is about the same topic as the query.

Query-Likelihood Model

- Rank documents by the probability that the query could be generated by the document model (i.e. same topic)
- Given query, start with P(D|Q)
- Using Bayes' Rule

$$p(D|Q) \overset{rank}{=} P(Q|D)P(D)$$

- Assuming priori is uniform, unigram model

$$P(Q|D) = \prod_{i=1}^{n} P(q_i|D)$$

Estimating Probabilities

- Obvious estimate for unigram probabilities is

$$P(q_i|D) = \frac{f_{q_i,D}}{|D|}$$

- If query words are missing from document, score will be zero
  - Missing 1 out of 4 query words same as missing 3 out of 4

Smoothing

- Document texts are a *sample* from the language model
  - Missing words should not have zero probability of occurring
- *Smoothing* is a technique for estimating probabilities for missing (or unseen) words
  - lower (or *discount*) the probability estimates for words that are seen in the document text
  - assign that "left-over" probability to the estimates for the words that are not seen in the text

Estimating Probabilities

- Estimate for unseen words is $\alpha_D P(q_i|C)$
  - $P(q_i|C)$ is the probability for query word *i* in the *collection* language model for collection *C* (background probability)
  - α_D is a parameter
- Estimate for words that occur is

$$(1 - \alpha_D)\, P(q_i|D) + \alpha_D\, P(q_i|C)$$

- Different forms of estimation come from different α_D

Jelinek-Mercer Smoothing

- α_D is a constant, λ
- Gives estimate of

$$p(q_i|D) = (1-\lambda)\frac{f_{q_i,D}}{|D|} + \lambda\frac{c_{q_i}}{|C|}$$

- Ranking score

$$P(Q|D) = \prod_{i=1}^{n}((1-\lambda)\frac{f_{q_i,D}}{|D|} + \lambda\frac{c_{q_i}}{|C|})$$

- Use logs for convenience
  - accuracy problems multiplying small numbers

$$\log P(Q|D) = \sum_{i=1}^{n}\log((1-\lambda)\frac{f_{q_i,D}}{|D|} + \lambda\frac{c_{q_i}}{|C|})$$

4. Relevance Models

- A relevance model represents the **topics** covered by relevant documents.
  - *E.g.,* language model can be used to represent information need
    - query and relevant documents are samples of text generated from this model
- Document likelihood model
  - $P(D|R)$ – is interpreted as the probability of generating the text in a document given a relevance model R.
  - less effective than query likelihood due to difficulties comparing across documents (often documents are large) of different lengths.
  - Note that a document with a model that is very similar to the relevance model is likely to be on the same topic.
  - how to compare two language models?

Pseudo-Relevance Feedback

- Relevance feedback is to acquire user feedback on the output that are initially returned from a given query *Q*.
- The feedback describes user information needs, and users typically label relevant outputs for *Q* (unlabelled outputs can be considered non-relevant information).
- In practical applications, there are three types of feedback: explicit feedback, implicit feedback, and "pseudo" feedback.
- **Pseudo relevance feedback** (blind feedback) is a method of finding an initial set of most likely relevant documents. Normally, we assume that the top "*k*" ranked documents are relevant to *Q*.
- Pseudo relevance feedback can be used to estimate *relevance model* from query *Q* and **top-k ranked documents.**
  - Then rank new documents by similarity of document model to this relevance model
  - *Kullback-Leibler divergence* (KL-divergence) is a well-known measure of the **difference** between two probability distributions. It can be used to measure the **similarity** between document model and relevance model.

KL-Divergence

- Given the *true* probability distribution *P* and another distribution *Q* that is an *approximation* to *P*,

$$KL(P||Q) = -\sum_x P(x) log \frac{P(x)}{Q(x)}$$

  - Use negative KL-divergence for ranking, and
  - assume relevance model *R for the query* is the true distribution (not symmetric), and the approximation to be the document language model (*D*):

$$\sum_{w \in V} P(w|R) \log P(w|D) - \sum_{w \in V} P(w|R) \log P(w|R)$$

Please note the second term of this equation does not depend on the document, and can be ignored for the purpose of ranking.

KL-Divergence cont.

- Given a simple maximum likelihood estimate for $P(w|R)$, based on the frequency in the query text, ranking score for a document is

$$\sum_{w \in V} \frac{f_{w,Q}}{|Q|} \log P(w|D)$$

  - rank-equivalent to query likelihood score
- Query likelihood model is a special case of retrieval based on relevance model.

Estimating the Relevance Model

- Probability of pulling a word *w* out of the "bucket" representing the relevance model depends on the *n* query words we have just pulled out

$$P(w|R) \approx P(w|q_1 \ldots q_n)$$

  - We view the probability of *w* is the conditional probability of observing *w* given that we just observed the query words $q_1 \ldots q_n$

- By definition

$$P(w|R) \approx \frac{P(w,q_1 \ldots q_n)}{P(q_1 \ldots q_n)}$$

Estimating the Relevance Model cont.

- Joint probability is

$$P(w, q_1 \ldots q_n) = \sum_{D \in \mathcal{C}} p(D) P(w, q_1 \ldots q_n | D)$$

- Assume

$$P(w, q_1 \ldots q_n | D) = P(w|D) \prod_{i=1}^{n} P(q_i|D)$$

- Gives

$$P(w, q_1 \ldots q_n) = \sum_{D \in \mathcal{C}} P(D) P(w|D) \prod_{i=1}^{n} P(q_i|D)$$

  - *Where P*(*D*) usually assumed to be uniform;
  - *P*(*w, $q_1 \ldots q_n$*) is simply a weighted average of the language model probabilities for *w* in a set of documents, where the weights are the query likelihood scores for those documents
  - Formal model for pseudo-relevance feedback
    - query expansion technique

Pseudo-Feedback Algorithm

1. Rank documents using the query likelihood score for query Q.
2. Select some number of the top-ranked documents to be the set C.
3. Calculate the relevance model probabilities $P(w|R)$ using the estimate for $P(w, q_1 \ldots q_n)$.
4. Rank documents again using the KL-divergence score:

$$\sum_w P(w|R) \log P(w|D)$$

Example from Top 10 Docs

president lincoln	*abraham lincoln*	*fishing*	*tropical fish*
lincoln	lincoln	fish	fish
president	america	farm	tropic
room	president	salmon	japan
bedroom	faith	new	aquarium
house	guest	wild	water
white	abraham	water	species
america	new	caught	aquatic
guest	room	catch	fair
serve	christian	tag	china
bed	history	time	coral
washington	public	eat	source
old	bedroom	raise	tank
office	war	city	reef
war	politics	people	animal
long	old	fishermen	tarpon
abraham	national	boat	fishery

Example from Top 50 Docs

president lincoln	*abraham lincoln*	*fishing*	*tropical fish*
lincoln	lincoln	fish	fish
president	president	water	tropic
america	america	catch	water
new	abraham	reef	storm
national	war	fishermen	species
great	man	river	boat
white	civil	new	sea
war	new	year	river
washington	history	time	country
clinton	two	bass	tuna
house	room	boat	world
history	booth	world	million
time	time	farm	state
center	politics	angle	time
kennedy	public	fly	japan
room	guest	trout	mile

Reference

- Chapter 7 (sections 7.1, 7.2 and 7.3) in text book - W. Bruce Croft, *Search Engines - Information retrieval in Practice*; Pearson, 2010.
- Albishre K., Li Y., Xu Y. (2018) *Query-Based Automatic Training Set Selection for Microblog Retrieval*. In: Phung D., Tseng V., Webb G., Ho B., Ganji M., Rashidi L. (eds) Advances in Knowledge Discovery and Data Mining. PAKDD 2018. Lecture Notes in Computer Science, vol 10938. **DOI** https://doi.org/10.1007/978-3-319-93037-4_26

- Next week Indexing Text and Ranking Documents – chapter 5 of text book.