

Statistical Language Modeling for Information Access

Theory II: Estimation, mixture models and semistructured documents

Maarten de Rijke Edgar Meij Krisztian Balog

University of Amsterdam
Norwegian University of Science and Technology

August 1–4, 2011

Outline

- ➊ A bit more on evaluation
- ➋ Modeling retrieval
- ➌ Mixture models and priors
- ➍ Applications to semistructured document retrieval

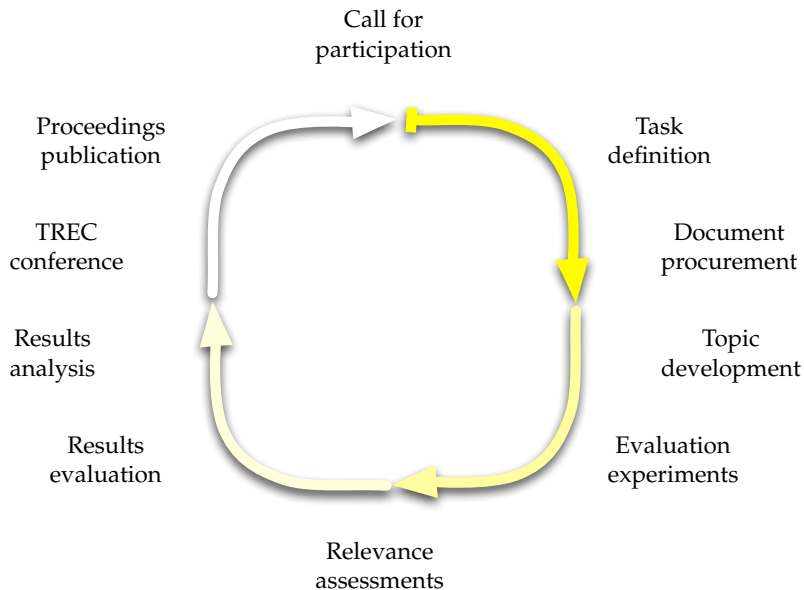
History

- Experimental methodology prominent in IR since 1960s
 - not sufficient to develop formalisms or approaches
 - **mandatory** to demonstrate effectiveness empirically
- Early work compared manual vs. automatic indexing
 - could automatic approach manual quality?
 - assumes that manual approach was the “correct” one
- Methods evolved to compare overall system performance
 - batch mode retrieval
 - interactive information seeking

TREC Conference

- Text REtrieval Conference
 - tasks, calls, proceedings at <http://trec.nist.gov>
- Established in 1991 to evaluate large-scale IR
 - retrieving documents from a gigabyte collection
- Organised by NIST and run continuously since 1991
 - TREC 2011 is in November, deadlines starting from August
- Best-known IR evaluation setting
 - started with 25 participating organizations in 1992
 - nowadays: 100+ groups from 20+ countries
 - European (CLEF) and Asian counterparts (NTCIR)
 - **CLEF 2011 held in Amsterdam, in September**
 - INEX
 - MUC, DUC
 - Example widely also widely followed in other areas
 - Senseval, MTEval, ...

TREC Cycle



TREC Cycle

- Most TREC tracks are organized as follows
 - November: track approved by TREC community
 - winter: track's members finalize format for track
 - spring: researchers train system based on specification
 - summer: researchers carry out formal evaluation
 - usually a “blind” evaluation: researchers do not know answer
 - fall: NIST carries out evaluation
 - November: Group meeting (TREC) to find out:
 - how well your site did
 - how others tackled the problem
 - some tracks are run by volunteers outside of NIST (e.g., Web)

- Widely recognized, premier annual IR evaluation
- What is good
 - brings together a wide range of active researches
 - huge distributed resources applied to common task
 - substantial gains on tasks rapidly
- What is less good
 - annual evaluation can divert resources from research
 - evaluations often require significant engineering effort
 - some tracks evaluation bi-annually as a result
 - recently, an explosion of tracks
 - means less energy applied to individual tasks

TREC Format

- TREC consists of IR research tracks
 - Past examples
 - ad-hoc retrieval, routing, cross-language, scanned documents, speech recognition, video, genomics, question answering, interactive, novelty, Web, NLP, robust, enterprise, blogs, SPAM
- TREC 2011
 - microblog, legal, session, medical, web, chemical, crowdsourcing, entity
- NLP spun off into TAC (Text Analysis Conference)
 - Knowledge base population task, Text entailment, Summarization

Outline

- ➊ A bit more on evaluation
- ➋ Modeling retrieval
- ➌ Mixture models and priors
- ➍ Applications to semistructured document retrieval

What is a Retrieval Model?

- Formal representation of the process of matching a query and a document
- Theory of relevance topical or user relevance
- Typically based on a statistical view of language
- Basis of a ranking algorithm
- Explicit or implicit

Retrieval Models

- Older models
 - Query languages, indexing (Boolean)
 - introducing ranking and weighting (Vector Space)
- Topical relevance models
 - IR as Bayesian classification, relevance information, tf.idf weights (BM25)
 - probabilistic models of documents, queries, topics (language models)
- More on Thursday

What makes it an language modeling technique?

- When is a given model an LM?
- LM is generative
 - at some level, an LM can be use to generate text
 - explicitly computes probability of observing a string of text
 - e.g., probability of obsserving a query string from a document model
 - model an entire population
- Discriminative approaches
 - model just the decision boundary
 - e.g., is this document relevant? does it belong to class X or Y?

LM-ing: pros and cons

- Pros
 - formal mathematical model
 - simple, well-understood framework
 - integrates both indexing and retrieval models
 - natural use of collection statistics, no heuristics
 - avoids “philosophical” concepts such as *relevance*, *aboutness*, etc.
- Cons
 - avoids “philosophical” concepts such as *relevance*, *aboutness*, etc.
 - relevance feedback, query expansion not straightforward
 - can’t easily accommodate phrases, passages, Boolean operators
- Extensions of LM overcome some issues

Issues in applying LMs

- What kind of LM should we use?
 - unigram or higher-order models?
 - multinomial or multiple Bernoulli?
- How can we estimate model parameters?
 - basic model
 - (translation models, aspect models, relevance models)
- How can we use the model for ranking?
 - query likelihood
 - document likelihood
 - (divergence of query and document models)

Unigram LMs

- Words are sampled independently from each other
 - “randomly pulling out words from an urn (with replacement)”
 - joint probability decomposes into a product of marginals
 - estimation of probabilities: simple counting
- Basic modeling: determine a posteriori most likely documents, i.e., for which $p(d|q)$ is highest:

$$p(d|q) = \frac{p(q|d) \cdot p(d)}{p(q)}$$
$$p(d|q) \propto p(q|d) \cdot p(d)$$

Query-likelihood term Query-independent term
term (often uniform)

Higher-Order Models

- Unigram model assumes word independence
 - cannot capture surface form
 - $P(\text{"brown fox"}) = P(\text{"fox brown"})$
- Higher-order models
 - n-gram: condition on preceding words
 - cache: condition on a window
 - grammar: condition on parse tree
- Are they useful?
 - no improvements from n-gram, grammar-based modules
 - some research on cache-like (proximity, passages, etc)
 - parameter estimation is prohibitively expensive

Higher-order Models

- Song and Croft, A general language model for information retrieval, *CIKM* 1999
- Combining unigrams with bigrams:
 - $p(t_{i-1}, t_i|d) = \lambda_1 \cdot p_1(t_i|d) + \lambda_2 \cdot p_2(t_{i-1}, t_i|d)$
 - $\lambda_1 + \lambda_2 = 1$
 - $p_2(t_1, t_2|d) = p_1(t_1|d) \cdot p_1(t_2|d, t_1)$
- Evaluation on the WSJ (250Mb, 74K docs) and TREC 4 (2Gb, 570K docs; WSJ \subseteq TREC 4) data sets

Higher-order Models

Table 3. Experimental Results on the WSJ Data Set

Retrieval Methods	11-pt Average	%Change	%Change
INQUERY	0.2172	-	
LM	0.2027	- 6.68%	-
GLM(40)	0.2198	+ 1.20%	+ 8.44%
GLM2(40+90)	0.2359	+ 8.61%	+ 16.38%

Table 4. Experimental Results on the TREC4 Data Set

Retrieval Methods	11-pt Average	%Change	%Change
INQUERY	0.1917	-	
LM	0.1890	- 1.41%	-
GLM(40)	0.1905	- 0.63%	+ 0.79%
GLM2(40+90)	0.1923	+ 0.31%	+ 1.75%

- Interesting improvements on small collection
- Negligible on more realistic collection sizes
- Findings corroborated in later work
- See tomorrow's lecture for an alternative way of modeling higher-order aspects (Gao et al.)

Multinomial or multiple-Bernoulli?

- Predominant model is the **multinomial**
 - Modeling word frequency
 - observation is a sequence of events, one for each query token
 - $P(t_1, \dots, t_k | M) = \prod_{i=1}^k P(t_i | M)$
- Some flavors are multiple-Bernoulli
 - Modeling word presence/absence
 - Observation is a vector of binary events, one for each possible word
 - $P(t_1, \dots, t_k | M) = \prod_{w \in t_1, \dots, t_k} P(w | M) \cdot \prod_{w \notin t_1, \dots, t_k} (1 - P(w | M))$

Multinomial or multiple-Bernoulli?

- Two models are fundamentally different
 - entirely different event spaces
 - both assume word independence (though it has different meanings)
 - both use smoothed relative-frequency (counting) for estimation
- Multinomial
 - can account for multiple word occurrences in the query
 - well understood
 - possibility for integration with ASR/MT/NLP (same event space)
- Multiple-Bernoulli
 - highly suited for IR (directly checks presence of query terms)
 - provision for explicit negation of query terms (“A but not B”)
 - no issues with observation length
- See Lavrenko *A General Theory of Relevance*, PhD thesis, UMass, 2004 for experimental assessment
 - Multinomial seems to work better

Ranking with LMs

- Standard approach: **query likelihood**
 - estimate language model M_D for every doc D in collection
 - rank docs by the probability of “generating” the query

$$P(q|M_D) = \prod_{t \in q}^k P(t|M_D)^{n(t,q)}$$

- Computation often performed in the **log** domain:

$$\log P(q|M_D) = \sum_{t \in q} n(t, q) \cdot \log P(t|M_D)$$

- Drawbacks
 - no notion of relevance in the model: everything is random sampling
 - user feedback/query expansion not part of the model
 - examples of relevant documents cannot help improve M_D
 - only option is augmenting the original query Q with extra terms
 - could, in principle, make use of sample queries for which D is relevant
 - does not directly allow weighted or structured queries

Ranking with LMs

- Document likelihood: flip the direction of the query likelihood approach
 - estimate a language model M_Q for the query Q
 - rank docs D by the likelihood of being random sample from M_Q
 - M_Q expected to “predict” a typical relevant document

$$P(D|M_Q) = \prod_{w \in D} P(w|M_Q)$$

- Drawbacks
 - different doc lengths, probabilities not comparable
 - favors docs that contain frequent (low content) words
 - consider “ideal” (highest-ranked) document for a given query

$$\max_D \prod_{w \in D} P(w|M_Q) = \max_{w \in D} P(w|M_Q)^n$$

Ranking with LMs

- Other choices in the literature
 - Likelihood ratio
 - “fix” document likelihood
 - related to probability ranking principle
 - Model comparison
 - estimate query model and doc model
 - use measure such as cross-entropy, KL-divergence to compare them
 - Hang on, let’s throw in some formulas... from query likelihood to KL divergence...

From Query Log-Likelihood to KL Divergence

- $p(q|M_d) = \prod_{t \in q} p(t|M_d)^{n(t,q)}$
- $\log p(q|M_d) = \sum_{t \in q} n(t,q) \cdot \log p(t|M_d)$
- Generalize $n(t,q)$ to $p(t|M_q)$:
 - $\log p(q|M_d) = \sum_{t \in q} p(t|M_q) \cdot \log p(t|M_d)$
- Recall KL-divergence: measuring the difference between two probability distributions:

$$\text{KL}(M_q \| M_d) = - \sum_t p(t|M_q) \log p(t|M_d) + \text{cons}(q)$$

- For those of you in the know: $\text{cons}(q)$ is document-independent, the entropy of the query model M_q
- $\text{cons}(q)$ does not affect the ranking of documents
- Hence, maximizing the query log-likelihood provides the same ranking as minimizing the KL-divergence

Constructing a Document Model

- So far: retrieval = unigram language model estimation problem
- How to infer a document model?
 - Represent d by a multinomial probability distribution over the vocabulary of terms, $p(t|d)$
 - Maximum likelihood estimate of a term given by relative frequency:

$$p(t|d) = \frac{n(t, d)}{n(d)}$$

- ... plus smoothing
 - Jelinek Mercer: linear interpolation with collection model

$$p(t|M_d) = (1 - \lambda) \cdot p(t|d) + \lambda \cdot p(t)$$

- Bayes smoothing aka Dirichlet smoothing

$$p(t|M_d) = \frac{n(t, d) + \beta \cdot p(t)}{n(d) + \beta}$$

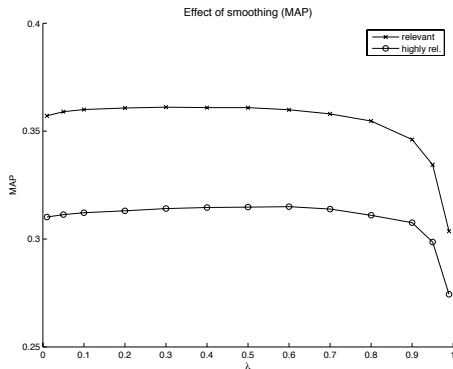
(β often set to average doc length)

Experimental Results on Smoothing

- TREC enterprise document retrieval task: finding relevant documents in an enterprise collection
- Balog, Weerkamp, de Rijke, A few Examples Go A Long Way: Constructing Query Models from Elaborate Query Formulations, *SIGIR* 2008
- To set up baseline:
 - empirically best
 - maximize average precision of a small set user-provided documents
 - maximize query likelihood, again using that small set of user provided documents

Experimental Results on Smoothing

- Note peculiarities in the paper
 - $p(t|M_d) = (1 - \lambda) \cdot p(t|d) + \lambda \cdot p(t)$
 - Jelinek-Mercer smoothing
- Empirical exploration:



Further remarks about smoothing

- Jelinek-Mercer and Dirichlet generally work well for IR
- Zhai and Lafferty (2002) consider a two-stage smoothing method
 - explain unseen words
 - explain noise in the query
 - $p(t|d) = (1 - \lambda) \frac{n(t,d) + \beta p(t)}{n(d) + \beta} + \lambda p(t|U)$
where U is a user background model, which can be approximated by $p(t|GE)$
- Parsimonious language models: instead of blindly modeling language use (through MLE and/or smoothing), model what language usage distinguishes a relevant document from other documents
 - See tomorrow

Outline

- ➊ A bit more on evaluation
- ➋ Modeling retrieval
- ➌ Mixture models and priors
- ➍ Applications to semistructured document retrieval

Exploiting Multiple Sources of Evidence

Some examples

- Document structure
 - Newspaper articles (lead, title)
 - HTML documents (content, meta tags, anchor texts)
- Collection structure
 - Digital library, with multiple media types, multiple collections, multiple journals, etc.
 - Expert finding, with multiple sources of evidence (e.g., publications, profiles, course material, annual reports, ...)

Query Independent Factors

Some examples

- Factors other than content-similarity that may/should influence document ranking
- Examples
 - Time: searching a news archive; prefer more recent items over old items
 - Credibility:
 - Link structure: use page rank to identify “authoritative” pages
 - Quality indicators: language usage, host
 - Opinionatedness (for marketing analysts): lexical scoring plus (perhaps #comments)
 - Expert finding: approachability, media experience?
 - Past search behavior? Past click behavior?

Putting Things Together

- Recall: baseline model

$$p(d|q) \propto p(d) \cdot p(q|d) = p(d) \cdot \prod_{t_i \in q} p(t_i|d)$$

- Multiple sources of evidence combined into a *mixture model*:

$$p(d|q) \propto p(d) \cdot \prod_{t_i \in q} ((1 - \lambda_1 - \dots - \lambda_k)p(t_i|S_{k+1}) + \lambda_1 p(t_i|S_1) + \dots + \lambda_k p(t_i|S_k))$$

- and we use $p(d)$ to model the *priors*
 - assume independence and use product for multiple priors

Outline

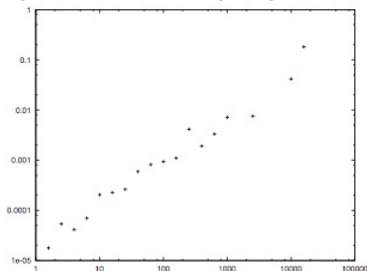
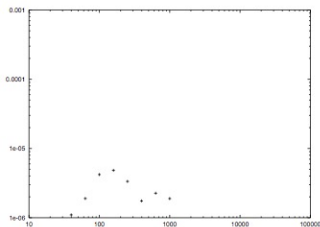
- ➊ A bit more on evaluation
- ➋ Modeling retrieval
- ➌ Mixture models and priors
- ➍ Applications to semistructured document retrieval

App 1: Entry Page Search

- Task: to find the home page of an institution (“the entry page”)
 - E.g., *Hunt Memorial Library*
- Priors
 - The prior probability of relevance vs doc length
 - The prior probability of relevance vs #inlinks
 - URL depth (“slash counting”)
- Sources of evidence
 - Anchor text
 - Content
- $p(d|q) \propto p(d) \prod_i ((1 - \lambda - \mu)p(t_i|GE) + \lambda p_{content}(t_i|d) + \mu p_{anchor}(t_i|d))$
- JM smoothing of $p_{content}$ and p_{anchor}

App 1: Entry Page Search

- Prior probabilities (doc length, #inlinks, log-log scales; url type)



URL type	Entry pages	WT10g
root	79 (73.1%)	12258 (0.7%)
subroot	15 (13.9%)	37959 (2.2%)
path	8 (7.4%)	83734 (4.9%)
file	6 (5.6%)	1557719 (92.1%)

App 1: Entry Page Search

- Kraaij, Westerveld, Hiemstra, Importance of Prior Probabilities for Entry Page Search, *SIGIR 2002*
 - TREC Web track 2001 data

Ranking method	Content ($\lambda = 0.9$)	Anchors($\lambda = 0.9$)	α	MRR
$P(Q D)$	0.3375	0.4188		
$P(Q D)P_{doclen}(D)$	0.2634	0.5600	0.5	0.3978
$P(Q D)P_{URL}(D)$	0.7705	0.6301	0.7	0.4703
$P(Q D)P_{inlink}(D)$	0.4974	0.5365	0.8	0.4920
			0.9	0.4797

Table 4: Results for different priors

Table 5: Combining web page text and anchor text

Ranking method	Content+Anchors ($\alpha = 0.8$)
$P(Q D)$	0.4920
$P(Q D)P_{URL}(D)$	0.7748
$P(Q D)P_{inlink}(D)$	0.5963

Table 6: Results for different priors(content+anchor)

App 2: Blog post retrieval

- Task: blog post retrieval

- given a topic, identify blog posts (“utterances”) that discuss the

	indicator	topic de- pendent?	post level/ blog level	related Rubin & Liddy indicator
--	-----------	-----------------------	---------------------------	------------------------------------

• E.	capitalization	no	post	4b	est in the topic pical Blog Post
• ≠	emoticons	no	post	4b	
• Weerk	shouting	no	post	4b	
Retrie	spelling	no	post	4b	
• Credik	post length	no	post	3a	lling, post
• Po	timeliness	yes	post	3d	
le	semantic	yes	post	3b, 3c	
• Bl	spam	no	blog	3b, 3c, 3f, 3g	
	comments	no	blog	1b	sistency,
	regularity	no	blog	2f	
	consistency	no	blog	2f	

Table 1: Credibility indicators

App 2: Blog post retrieval

- Modeling things...
- Priors (topic independent!)
 - $p(d) = \lambda \cdot p_{pl}(d) + (1 - \lambda) \cdot p_{bl}(d)$
 - $p_{pl}(d) = \sum_i \frac{1}{5} \cdot p_i(d)$
 - $p_{bl}(d) = \sum_i \frac{1}{4} \cdot p_i(d)$
- Topic dependent indicators: create a query model that mixes a temporal query model and a semantic query model:
 - $p(t|M_q) = \mu \cdot p(t|M_{temporal}) + (1 - \mu) \cdot p(t|M_{semantic})$
- Putting it all together
 - $\log p(d|q) \propto \beta(\sum_t p(t|q) \cdot \log p(t|M_d)) + (1 - \beta)(\sum_t p(t|M_q) \cdot \log p(t|M_d))$
- For details on estimation see paper on course wiki

App 2: BI

- But doe
- TREC B
- Develop

	2006		2007	
	map	p@10	map	p@10
baseline	0.2156	0.4360	0.2820	0.5160
capitalization	0.2155	0.4500	0.2824	0.5160
emoticons	0.2156	0.4360	0.2820	0.5200
shouting	0.2159	0.4320	0.2833	0.5100
spelling	0.2179 [△]	0.4480 [△]	0.2839 [▲]	0.5220
post length	0.2502 [▲]	0.4960 [▲]	0.3112 [▲]	0.5700 [▲]
timeliness	0.1865 [▼]	0.4520	0.2660	0.4860
semantic	0.2840 [▲]	0.6240 [▲]	0.3379 [▲]	0.6640 [▲]
spam filtering	0.2093	0.4700	0.2814	0.5760 [▲]
comments	0.2497 [▲]	0.5000 [▲]	0.3099 [▲]	0.5600 [▲]
regularity	0.1658 [▼]	0.4940 [△]	0.2353 [▼]	0.5640 [△]
consistency	0.2141 [▼]	0.4220	0.2785 [▼]	0.5040
post level (topic indep.)	0.2374 [▲]	0.4920 [▲]	0.2990 [▲]	0.5660 [▲]
post level (topic dep.)	0.2840 [▲]	0.6240 [▲]	0.3379 [▲]	0.6640 [▲]
post level (all)	0.2911 [▲]	0.6380 [▲]	0.3369 [▲]	0.6620 [▲]
blog level	0.2391 [▲]	0.4500	0.3023 [▲]	0.5580 [▲]
all	0.3051 [▲]	0.6880 [▲]	0.3530 [▲]	0.6900 [▲]

Table 2: Retrieval performance on 2006 and 2007 topics, using $\lambda = 0.3$, $\beta = 0.4$, and $\mu = 0.0$.

Wrap Up and Look Ahead

- The course web site
- Summary
 - A bit on **evaluation**
 - **Revisiting basic language modeling** for IR
 - **Mixture models and priors** to incorporate document structure and aspects that go beyond relevance
- Tomorrow
 - Incorporating symbolic knowledge
 - Cluster-based LMs
 - Applications to biomedical IR and more
- On to today's practical part