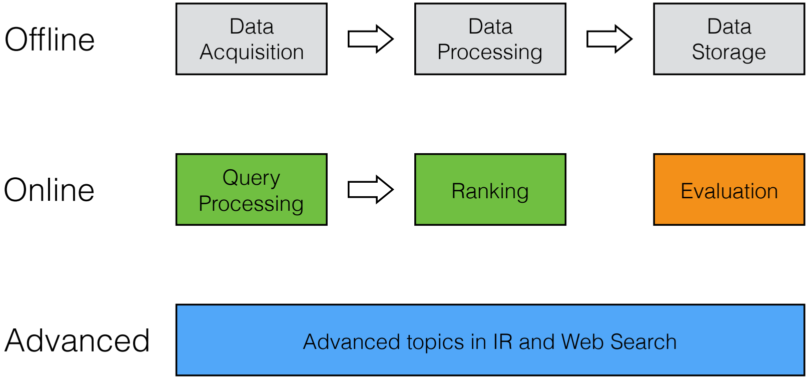
Information Retrieval

Term-based Retrieval

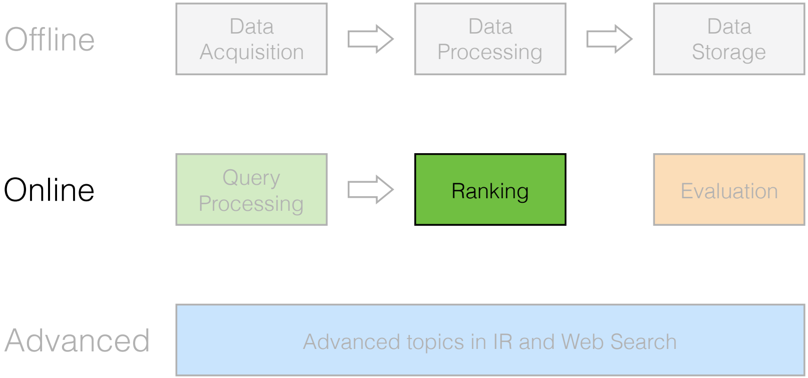
Ilya Markov
i.markov@uva.nl

University of Amsterdam

Course overview



Next few lectures



Ranking methods

1. Content-based
   - **Term-based**
   - Semantic
2. Link-based (web search)
3. Learning to rank

Outline

Outline

1. Vector space model
   - Method
   - Relevance feedback
2. Probabilistic IR
3. Language modeling in IR

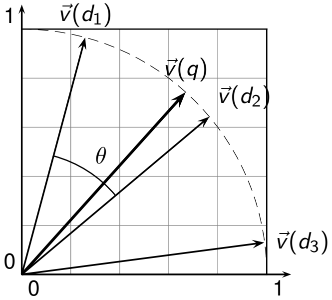
Outline

Documents as vectors

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth	...
Anthony	1	1	0	0	0	1	
Brutus	1	1	0	1	0	0	
Caesar	1	1	0	1	1	1	
Calpurnia	0	1	0	0	0	0	
Cleopatra	1	0	0	0	0	0	
mercy	1	0	1	1	1	1	
worser	1	0	1	1	1	0	
...							

Manning et al., "Introduction to Information Retrieval"

Vector space model



$$sim(d,q) = \cos(\vec{v}(d), \vec{v}(q)) = \frac{\vec{v}(d) \cdot \vec{v}(q)}{\|\vec{v}(d)\| \cdot \|\vec{v}(q)\|}$$

$$= \frac{\sum_{i=1}^{|V|} d_i \cdot q_i}{\sqrt{\sum_{i=1}^{|V|} d_i^2} \cdot \sqrt{\sum_{i=1}^{|V|} q_i^2}}$$

Manning et al., "Introduction to Information Retrieval"

Term frequency

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth	...
Anthony	157	73	0	0	0	1	
Brutus	4	157	0	2	0	0	
Caesar	232	227	0	2	1	0	
Calpurnia	0	10	0	0	0	0	
Cleopatra	57	0	0	0	0	0	
mercy	2	0	3	8	5	8	
worser	2	0	1	1	1	5	
...							

Manning et al., "Introduction to Information Retrieval"

Term frequency

Raw term frequency	$tf(t, d)$
Log term frequency	$\begin{cases} 1 + \log tf(t, d) & \text{if } tf(t, d) > 0 \\ 0 & \text{otherwise} \end{cases}$

Inverse document frequency

$$idf(t) = \log \frac{N}{df(t)}$$

- $df(t)$ – document frequency of term t
- N – total number of documents in a collection

Inverse document frequency

Term	$df(t)$	$idf(t)$
calpurnia	1	6
animal	100	4
sunday	1000	3
fly	10,000	2
under	100,000	1
the	1,000,000	0

for $N = 1,000,000$ and $\log_{10}$

Manning et al., "Introduction to Information Retrieval"

TF-IDF

$$\text{TF-IDF}(t, d) = tf(t, d) \cdot idf(t)$$

- Term frequency
  - $tf(t, d)$
  - $\begin{cases} 1 + \log tf(t, d) & \text{if } tf(t, d) > 0 \\ 0 & \text{otherwise} \end{cases}$
- Inverse document frequency
  - $\log \frac{N}{df(t)}$
  - $\max\{0, \log \frac{N - df(t)}{df(t)}\}$

Vector space model summary

- Documents and queries as vectors
- Rank documents using cosine similarity
- Weights can be
  1. binary
  2. term frequency
  3. TF-IDF

Outline

Relevance feedback

1. The user issues a (short, simple) query
2. The system returns an initial set of retrieval results
3. Some returned results are identified as relevant or non-relevant
4. The system computes a better representation of the information need based on this feedback
5. The system displays a revised set of retrieval results

Relevance feedback in VSM

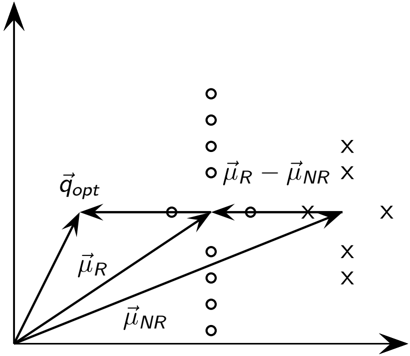
- D_r, D_{nr} – sets of relevant and non-relevant documents
- $\mu(D_r)$, $\mu(D_{nr})$ – vector centroids of the corresponding sets
- Rocchio algorithm

$$\vec{q}_{opt} = \operatorname*{argmax}_{\vec{q}}[sim(\vec{q}, \mu(D_r)) - sim(\vec{q}, \mu(D_{nr}))]$$

- Approximated as

$$\begin{aligned}\vec{q}_{opt} &= \mu(D_r) + [\mu(D_r) - \mu(D_{nr})] \\ &= \frac{1}{|D_r|}\sum_{\vec{d}_j \in D_r} \vec{d}_j + \left[\frac{1}{|D_r|}\sum_{\vec{d}_j \in D_r} \vec{d}_j - \frac{1}{|D_{nr}|}\sum_{\vec{d}_j \in D_{nr}} \vec{d}_j\right]\end{aligned}$$

Rocchio algorithm



Manning et al., "Introduction to Information Retrieval"

Rocchio algorithm in practice

$$\vec{q}_{opt} = \alpha \vec{q}_0 + \beta \mu(D_r) - \gamma \mu(D_{nr})$$
$$= \alpha \vec{q}_0 + \beta \frac{1}{|D_r|} \sum_{\vec{d}_j \in D_r} \vec{d}_j - \gamma \frac{1}{|D_{nr}|} \sum_{\vec{d}_j \in D_{nr}} \vec{d}_j$$

- More judged documents $\Rightarrow$ higher values of β and γ
- Reasonable values are $\alpha = 1, \beta = 0.75, \gamma = 0.15$

Summary

1. Vector space model
    - Method
    - Relevance feedback

Outline

Outline

2 Probabilistic IR
- Probability theory and statistics
- Method
- Relevance feedback
- Intermezzo: experimental comparison
- BM25

Basic probability theory

- For events A and B
  - Joint probability $P(A \cap B)$ of both events occurring
  - Conditional probability $P(A \mid B)$ of event A occurring given that event B has occurred
- Chain rule

$$P(A, B) = P(A \cap B) = P(A \mid B)P(B) = P(B \mid A)P(A)$$

- Partition rule: partition $P(B)$ based on A and $\overline{A}$

$$P(B) = P(A, B) + P(\overline{A}, B)$$

Manning et al., "Introduction to Information Retrieval"

Bayes' rule

$$\overbrace{P(A \mid B)}^{\text{posterior}} = \frac{\overbrace{P(B \mid A)}^{\text{likelihood}}\ \overbrace{P(A)}^{\text{prior}}}{P(B)} = \left[\frac{P(B \mid A)}{\sum_{X \in \{A, \bar{A}\}} P(B, X)}\right] P(A)$$

$$= \left[\frac{P(B \mid A)}{\sum_{X \in \{A, \bar{A}\}} P(B \mid X) P(X)}\right] P(A)$$

- $P(A)$ – prior probability, i.e., the initial estimate of how likely the event A is in the absence of any other information
- $P(B \mid A)$ – likelihood of the evidence B given the model A
- $P(A \mid B)$ – posterior probability of A after having seen the evidence B

Manning et al., "Introduction to Information Retrieval"

Odds

$$O(A) = \frac{P(A)}{P(\overline{A})} = \frac{P(A)}{1 - P(A)}$$

Conjugate prior

$$\overbrace{p(\theta \mid x)}^{posterior} = \frac{\overbrace{p(x \mid \theta)}^{likelihood}\ \overbrace{p(\theta)}^{prior}}{\int p(x \mid \theta')p(\theta')d\theta'}$$

- The likelihood function $p(x \mid \theta)$ is usually well-determined from a statement of the data-generating process
- For certain choices of the prior distribution $p(\theta)$, the posterior distribution $p(\theta \mid x)$ is in the same family of distributions
- Such distribution $p(\theta)$ is a conjugate prior for the likelihood function $p(x \mid \theta)$

https://en.wikipedia.org/wiki/Conjugate_prior

Conjugate prior for Bernoulli and binomial

- Bernoulli distribution
  - A random variable takes the value 1 with success probability p and the value 0 with failure probability $q = 1 - p$
- Binomial distribution
  - The number of successes in a sequence of n independent yes/no experiments, each of which yields success with probability p (Bernoulli trial)
- Beta distribution – conjugate prior for Bernoulli and binomial

$$\frac{x^{\alpha-1}(1-x)^{\beta-1}}{B(\alpha, \beta)}$$

Conjugate prior for Bernoulli and binomial

- Consider $n = s + f$ Bernoulli trials with success probability p
- Likelihood function

$$\mathcal{L}(s, f \mid p = x) = \binom{s+f}{s} x^s (1-x)^f = \binom{n}{s} x^s (1-x)^{n-s}$$

- Prior probability

$$P_{prior}(p = x; \alpha_{pr}, \beta_{pr}) = \frac{x^{\alpha_{pr}-1}(1-x)^{\beta_{pr}-1}}{B(\alpha_{pr}, \beta_{pr})}$$

- Posterior probability

$$P_{post}(p = x \mid s, f) = \frac{Prior(p = x; \alpha_{pr}, \beta_{pr})\mathcal{L}(s, f \mid p = x)}{\int_0^1 Prior(p = x; \alpha_{pr}, \beta_{pr})\mathcal{L}(s, f \mid p = x)dx}$$

https://en.wikipedia.org/wiki/Beta_distribution#Bayesian_inference

Conjugate prior for Bernoulli and binomial

$$
\begin{aligned}
P_{post}(p = x \mid s, f) &= \frac{1}{\mathcal{Z}} Prior(p = x; \alpha_{pr}, \beta_{pr}) \cdot \mathcal{L}(s, f \mid p = x) \\
&= \frac{1}{\mathcal{Z}} \binom{n}{s} x^s (1-x)^{n-s} \cdot \frac{x^{\alpha_{pr}-1}(1-x)^{\beta_{pr}-1}}{B(\alpha_{pr}, \beta_{pr})} \\
&= \frac{1}{\mathcal{Z}} \binom{n}{s} \frac{x^{s+\alpha_{pr}-1}(1-x)^{n-s+\beta_{pr}-1}}{B(\alpha_{pr}, \beta_{pr})} \\
&= \frac{\binom{n}{s} x^{s+\alpha_{pr}-1}(1-x)^{n-s+\beta_{pr}-1} / B(\alpha_{pr}, \beta_{pr})}{\int_0^1 \left(\binom{n}{s} x^{s+\alpha_{pr}-1}(1-x)^{n-s+\beta_{pr}-1} / B(\alpha_{pr}, \beta_{pr})\right) dx} \\
&= \frac{x^{s+\alpha_{pr}-1}(1-x)^{n-s+\beta_{pr}-1}}{\int_0^1 \left(x^{s+\alpha_{pr}-1}(1-x)^{n-s+\beta_{pr}-1}\right) dx} \\
&= \frac{x^{s+\alpha_{pr}-1}(1-x)^{n-s+\beta_{pr}-1}}{B(s+\alpha_{pr}, n-s+\beta_{pr})} \\
&\sim Beta(s+\alpha_{pr}, n-s+\beta_{pr})
\end{aligned}
$$

Conjugate prior for multinomial

- Multinomial distribution
  - The probability of counts for rolling a k-sided dice n times
  - Probability mass function

$$\mathcal{L}(n_1, \ldots, n_k \mid p_1 = x_1, \ldots, p_k = x_k) = \frac{n!}{n_1! \ldots n_k!} x_1^{n_1} \ldots x_k^{n_k}$$

  - Bernoulli is multinomial with $k = 2, n = 1$
  - Binomial is multinomial with $k = 2$
- Dirichlet distribution – conjugate prior for multinomial

$$P_{prior}(p_1 = x_1, \ldots, p_k = x_k; \alpha_1^{pr}, \ldots, \alpha_k^{pr}) = \frac{1}{B(\boldsymbol{\alpha})} \prod_{i=1}^{k} x_i^{\alpha_i^{pr} - 1}$$

  - Beta is Dirichlet with $k = 2$
- Posterior

$$P_{post}(p_1 = x_1, \ldots, p_k = x_k \mid n_1, \ldots, n_k) = \frac{1}{B(\boldsymbol{\alpha} + \mathbf{n})} \prod_{i=1}^{k} x_i^{\alpha_i^{pr} + n_i - 1}$$

Summary

- Probability theory
  - Bayes' rule
  - Odds
- Statistics
  - Conjugate priors

Outline

Probability ranking principle (PRP)

- Consider binary relevance $R \in \{0, 1\}$ and the probability of relevance $P(R = 1 \mid d, q)$
- PRP in brief

> If the retrieved documents d w.r.t. a query q are ranked decreasingly on their probability of relevance $P(R = 1 \mid d, q)$, then the effectiveness of the system will be the best that is obtainable.

- The relevance of each document is independent of the relevance of other documents

Manning et al., "Introduction to Information Retrieval"

Binary independence model (BIM)

- **Binary** (equivalent to Boolean): documents and queries are represented as binary term incidence vectors
- **Independence**: no association between terms

Manning et al., "Introduction to Information Retrieval"

Binary incidence matrix

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth	...
Anthony	1	1	0	0	0	1	
Brutus	1	1	0	1	0	0	
Caesar	1	1	0	1	1	1	
Calpurnia	0	1	0	0	0	0	
Cleopatra	1	0	0	0	0	0	
mercy	1	0	1	1	1	1	
worser	1	0	1	1	1	0	
...							

Manning et al., "Introduction to Information Retrieval"

Ranking under BIM

- Represent documents d and queries q as vectors $\vec{x}$ and $\vec{q}$
- Rank documents by the probability of relevance w.r.t. a query $P(R = 1 \mid \vec{x}, \vec{q})$
- Rank documents by odds $O(R \mid \vec{x}, \vec{q}) = \frac{P(R=1|\vec{x},\vec{q})}{P(R=0|\vec{x},\vec{q})}$

Computing odds

$$O(R \mid \vec{x}, \vec{q}) = \frac{P(R=1 \mid \vec{x}, \vec{q})}{P(R=0 \mid \vec{x}, \vec{q})} = \frac{\frac{P(R=1|\vec{q})P(\vec{x}|R=1,\vec{q})}{P(\vec{x}|\vec{q})}}{\frac{P(R=0|\vec{q})P(\vec{x}|R=0,\vec{q})}{P(\vec{x}|\vec{q})}}$$

$$= \frac{P(R=1 \mid \vec{q})}{P(R=0 \mid \vec{q})} \cdot \frac{P(\vec{x} \mid R=1, \vec{q})}{P(\vec{x} \mid R=0, \vec{q})}$$

$$\stackrel{\text{rank}}{=} \frac{P(\vec{x} \mid R=1, \vec{q})}{P(\vec{x} \mid R=0, \vec{q})}$$

Computing odds (cont'd)

$$O(R \mid \vec{x}, \vec{q}) \stackrel{\text{rank}}{=} \frac{P(\vec{x} \mid R = 1, \vec{q})}{P(\vec{x} \mid R = 0, \vec{q})} = \prod_{t=1}^{M} \frac{P(x_t \mid R = 1, \vec{q})}{P(x_t \mid R = 0, \vec{q})}$$

$$= \prod_{t:x_t=1} \frac{P(x_t = 1 \mid R = 1, \vec{q})}{P(x_t = 1 \mid R = 0, \vec{q})} \cdot \prod_{t:x_t=0} \frac{P(x_t = 0 \mid R = 1, \vec{q})}{P(x_t = 0 \mid R = 0, \vec{q})}$$

$$= \prod_{t:x_t=1} \frac{p_t}{u_t} \cdot \prod_{t:x_t=0} \frac{1 - p_t}{1 - u_t}$$

Computing odds (cont'd)

- $p_t = P(x_t = 1 \mid R = 1, \vec{q})$ – the probability of a term appearing in a relevant document
- $u_t = P(x_t = 1 \mid R = 0, \vec{q})$ – the probability of a term appearing in a non-relevant document

		Doc. rel. ($R = 1$)	Doc. non-rel. ($R = 0$)
Term present	$x_t = 1$	p_t	u_t
Term absent	$x_t = 0$	$1 - p_t$	$1 - u_t$

- Assume that if $q_t = 0$, then $p_t = u_t$

Manning et al., "Introduction to Information Retrieval"

Computing odds (cont'd)

$$
\begin{aligned}
O(R \mid \vec{x}, \vec{q}) &\overset{\text{rank}}{=} \prod_{t:x_t=q_t=1} \frac{p_t}{u_t} \cdot \prod_{t:x_t=0,q_t=1} \frac{1-p_t}{1-u_t} \\
&= \prod_{t:x_t=q_t=1} \frac{p_t(1-u_t)}{u_t(1-p_t)} \cdot \prod_{t:q_t=1} \frac{1-p_t}{1-u_t} \\
&\overset{\text{rank}}{=} \prod_{t:x_t=q_t=1} \frac{p_t(1-u_t)}{u_t(1-p_t)}
\end{aligned}
$$

Retrieval status value (RSV)

- Retrieval status value

$$RSV_d = \log \prod_{t:x_t=q_t=1} \frac{p_t(1-u_t)}{u_t(1-p_t)} = \sum_{t:x_t=q_t=1} \log \frac{p_t(1-u_t)}{u_t(1-p_t)}$$

- Log odds ratio

$$c_t = \log \frac{p_t(1-u_t)}{u_t(1-p_t)} = \log \frac{p_t}{1-p_t} - \log \frac{u_t}{1-u_t}$$

- $RSV_d = \sum_{t:x_t=q_t=1} c_t$
- Similar to VSM with c_t as term weights

Computing p_t and u_t

		Doc. rel.	Doc. non-rel.	Total
Term present	$x_t = 1$	s	$df(t) - s$	$df(t)$
Term absent	$x_t = 0$	$S - s$	$[N - df(t)] - [S - s]$	$N - df(t)$
	Total	S	$N - S$	N

$$p_t = \frac{s}{S}$$

$$u_t = \frac{df(t) - s}{N - S}$$

$$c_t = \log \frac{s}{S - s} - \log \frac{df(t) - s}{[N - df(t)] - [S - s]} \approx \log \frac{s}{S - s} - \log \frac{df(t)}{N - df(t)}$$

Manning et al., "Introduction to Information Retrieval"

Probabilistic IR summary

- Probability ranking principle (PRP)
  - Rank documents by $P(R = 1 \mid d, q)$
  - Need to estimate $P(R = 1 \mid d, q)$
- Binary independence model (BIM)
  - Binary representation of documents/queries/relevance
  - Terms are independent
- Retrieval status value

$$RSV_d = \sum_{t: x_t = q_t = 1} \log \frac{p_t(1 - u_t)}{u_t(1 - p_t)}$$

- Computing p_t and u_t

$$p_t = \frac{s}{S}, \;\; u_t = \frac{df(t) - s}{N - S}$$

Outline

2. Probabilistic IR
   - Probability theory and statistics
   - Method
   - Relevance feedback
   - Intermezzo: experimental comparison
   - BM25

Relevance feedback in probabilistic retrieval

1. Guess initial estimates of p_t and u_t
2. Rank results by RSV
3. Suppose a user judged V results, where $VR = \{d \in V : R_{dq} = 1\}$
4. If VR is large enough, then reestimate p_t and u_t

$$p_t = \frac{VR_t}{VR}, \;\; u_t = \frac{df(t) - VR_t}{N - VR}$$

5. Repeat from step 2

Manning et al., "Introduction to Information Retrieval"

Relevance feedback in probabilistic retrieval

- VR is usually small
- Use Bayesian estimation via conjugate priors
- The distribution of p_t and u_t is Bernoulli
- The conjugate prior is beta
- The Bayesian estimate for p_t (u_t is similar):

$$p_t^{(t+1)} = \frac{|VR_t| + \kappa p_t^{(k)}}{|VR| + \kappa}$$

- Why do we need κ?

Outline

2. Probabilistic IR
   - Probability theory and statistics
   - Method
   - Relevance feedback
   - Intermezzo: experimental comparison
   - BM25

Content-based retrieval methods

run	precision at recall: 0.2	0.5	0.8	average precision
tfc.tfc	0.211	0.100	0.026	0.126
probabilistic	0.247	0.185	0.079	0.165
Lnu.ltu	0.365	0.227	0.065	0.229
BM25	0.392	0.242	0.073	0.243
LM	0.428	0.265	0.130	0.277

D. Hiemstra and A. de Vries, "Relating the new language models of information retrieval to the traditional retrieval models"

Improvements that don't add up (baselines)

- Over half the baseline scores are below the median score (TREC systems in 1999)
- Only four baselines are in the top quartile
- Only one baseline is close to the best of the original TREC 1999 submissions
- The mean baseline score prior to 2005 is 0.260; from 2005 onwards it is 0.245

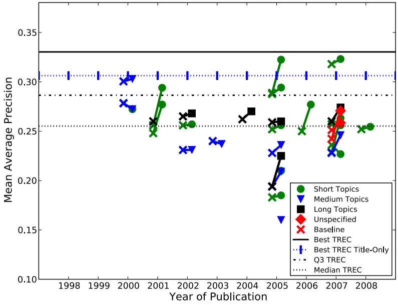


Figure: TREC-8 Ad-Hoc

T. Armstrong et al., "Improvements That Don't Add Up: Ad-Hoc Retrieval Results Since 1998"

Improvements that don't add up (improvements)

- The improved scores do not trend upwards over time
- Only five of the 30 improved scores are in the top quartile
- Only two title-only systems beat the best automatic TREC 1999 title-only system
- No system beats the best automatic TREC 1999 system across all query types

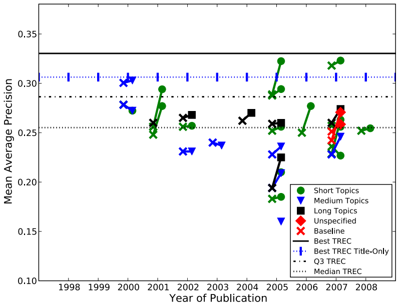


Figure: TREC-8 Ad-Hoc

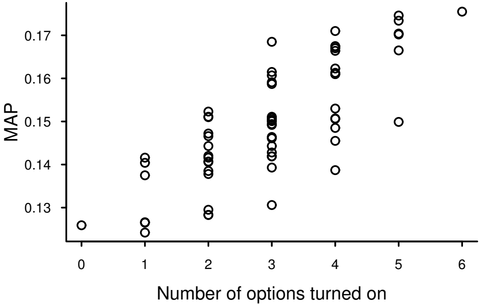
T. Armstrong et al., "Improvements That Don't Add Up: Ad-Hoc Retrieval Results Since 1998"

Additivity of improvements

Toggle	Enabled	Disabled
Term Smoothing	Dirichlet Prior [Zhai and Lafferty, 2004].	Jelinek-Mercer.
Ordered Phrases	Ordered proximity windows, with a maximum of 4 terms between each occurence, scored for every sequence of 2 or 3 terms in the original query [Metzler and Croft, 2005]. Tuning resulted in a weighting of 0.1/1.0.	No ordered proximity.
Unordered Proximity	Unordered proximity windows, with a maximum size of four times the number of terms being scored, for every sequence of two or three terms in the original query [Metzler and Croft, 2005] (This diverges slightly from the original method. described in the paper, but the number of possible combinations grows exponentially with query length). Tuning resulted in a weighting of 0.1/1.0.	No unordered proximity.
Query Expansion	Pseudo relevance feedback, using Indri's adapted version of relevance modelling [Lavrenko and Croft, 2001] with a total of twenty terms selected from ten documents, weighting the original query as 0.3 and the expanded query 0.7.	No query expansion.
Stemming	Porter Stemming.	No stemming.
Stopping	Stopping using the standard list of 417 stopwords included in Indri.	No stopping.

T. Armstrong et al., "Improvements That Don't Add Up: Ad-Hoc Retrieval Results Since 1998"

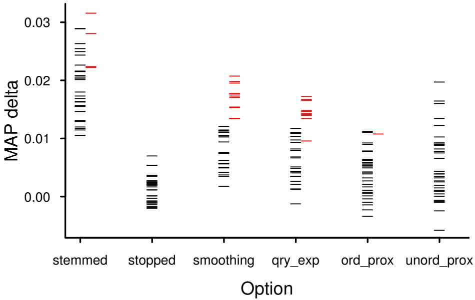
Additivity of improvements



- There is a positive relationship between the number of options turned on and the retrieval effectiveness achieved
- Options are broadly additive

T. Armstrong et al., "Improvements That Don't Add Up: Ad-Hoc Retrieval Results Since 1998"

Additivity of improvements



- The improvement, that an option offers, depends upon the combination of other options
- The improvements are highly variable

T. Armstrong et al., "Improvements That Don't Add Up: Ad-Hoc Retrieval Results Since 1998"

Outline

2 Probabilistic IR
- Probability theory and statistics
- Method
- Relevance feedback
- Intermezzo: experimental comparison
- BM25

Probabilistic retrieval revisited

- Assumptions
  - Boolean representation of documents/queries/relevance
  - Term independence
  - Out-of-query terms do not affect retrieval
  - Document relevance values are independent
- Similar to VSM
- But does not consider the term frequency and document length

BM25

- Start with a simple RSV

$$RSV_d = \sum_{t \in q} \log \left[\frac{N}{df(t)} \right]$$

- Factor in the term frequency and document length

$$RSV_d = \sum_{t \in q} \log \left[\frac{N}{df(t)} \right] \cdot \frac{(k_1 + 1) \cdot tf(t, d)}{k_1 \cdot \left[(1 - b) + b \cdot \frac{dl(d)}{dl_{ave}} \right] + tf(t, d)}$$

- k_1, b – parameters
- $dl(d)$ – length of document d
- dl_{ave} – average document length

BM25

$$BM25_d = \sum_{t \in q} \log\left[\frac{N}{df(t)}\right] \cdot \frac{(k_1 + 1) \cdot tf(t,d)}{k_1 \cdot \left[(1-b) + b \cdot \frac{dl(d)}{dl_{ave}}\right] + tf(t,d)}$$

- What if $k_1 \in \{0, 1, \infty\}$?
- What of $b \in \{0, 1\}$?
- What if $tf(t,d)$ is small/large? $k_1 \in [1.2, 2], b = 0.75$.

BM25 for long queries

$$BM25_d = \sum_{t \in q} \log \left[\frac{N}{df(t)} \right] \cdot \frac{(k_1 + 1) \cdot tf(t,d)}{k_1 \cdot \left[(1 - b) + b \cdot \frac{dl(d)}{dl_{ave}} \right] + tf(t,d)} \cdot \frac{(k_3 + 1) tf(t,q)}{k_3 + tf(t,q)}$$

Relevance feedback for BM25

$$BM25_d = \sum_{t \in q} \log \left[\frac{N}{df(t)} \right] \cdot \frac{(k_1 + 1) \cdot tf(t,d)}{k_1 \cdot \left[(1-b) + b \cdot \frac{dl(d)}{dl_{ave}} \right] + tf(t,d)}$$

- Use log odds instead

$$c_t = \log \frac{p_t(1-u_t)}{u_t(1-p_t)}$$

- Estimate p_t and u_t through relevance feedback

$$p_t = \frac{VR_t}{VR}, \quad u_t = \frac{df(t) - VR_t}{N - VR}$$

- Plug p_t and u_t into c_t and then c_t into $BM25_d$

$$c_t = \log \frac{|VR_t|/|VNR_t|}{[df(t) - |VR_t|]/[(N - |VR|) - (df(t) - |VR_t|)]}$$

Summary

2. Probabilistic IR
    - Probability theory and statistics
    - Method
    - Relevance feedback
    - Intermezzo: experimental comparison
    - BM25

Outline

Outline

3 Language modeling in IR
- Method
- Relevance feedback
- Smoothing

Language model

A statistical language model is a probability distribution over sequences of words.

- Given a sequence of length m
- A language model assigns probability $P(w_1, \ldots, w_m)$ to this sequence
- Unigram language model

$$P(w_1, \ldots, w_m) = P(w_1) \ldots P(w_m)$$

- Bi-gram language model

$$P(w_1, \ldots, w_m) = P(w_1)P(w_2 \mid w_1)P(w_3 \mid w_2) \ldots P(w_m \mid w_{m-1})$$

https://en.wikipedia.org/wiki/Language_model

Unigram language model example

Model M_1		Model M_2	
the	0.2	the	0.15
a	0.1	a	0.12
frog	0.01	frog	0.0002
toad	0.01	toad	0.0001
said	0.03	said	0.03
likes	0.02	likes	0.04
that	0.04	that	0.04
dog	0.005	dog	0.01
cat	0.003	cat	0.015
monkey	0.001	monkey	0.002
...	...	...	...

Manning et al., "Introduction to Information Retrieval"

Query likelihood model

- Rank documents by their likelihood given a query

$$P(d \mid q) = \frac{P(q \mid d)P(d)}{P(q)}$$

- The prior distribution over queries $P(q)$ does not affect the ranking for a particular query

$$P(d \mid q) \stackrel{rank}{=} P(q \mid d)P(d)$$

- Usually, the prior distribution over documents $P(d)$ is assumed to be uniform

$$P(d \mid q) \stackrel{rank}{=} P(q \mid d)$$

- $P(q \mid d) = P(q \mid M_d)$ is the probability that the query q is generated by the document language model M_d

Estimating query likelihood

- "Bag of words" assumption: terms are independent

$$P(q \mid M_d) = \prod_{t \in q} P(t \mid M_d)$$

- Unigram language model

$$P(t \mid M_d) = \frac{tf(t, d)}{dl(d)}$$

- If some query terms do not appear in document d, then $P(q \mid M_d) = 0$
- This is addressed by smoothing (discussed later)

Outline

Relevance model

- Assume there is an oracle language model M_r, called the *relevance model*
- Kullback-Leibler divergence between M_r and M_d

$$
\begin{aligned}
KL(M_r \| M_d) &= \sum_{t \in V} P(t \mid M_r) \log \frac{P(t \mid M_r)}{P(t \mid M_d)} \\
&= \sum_{t \in V} [P(t \mid M_r) \log P(t \mid M_r) - P(t \mid M_r) \log P(t \mid M_d)] \\
&\overset{rank}{=} - \sum_{t \in V} P(t \mid M_r) \log P(t \mid M_d)
\end{aligned}
$$

Estimating relevance model

- If we assume that the relevance model M_r is the query language model M_q, then

$$P(t \mid M_r) = \frac{tf(t, q)}{|q|}$$

- The out-of-query terms do not contribute to the KL score
- If we assume that query terms are sampled from the relevance model M_r, then

$$P(t \mid M_r) \approx P(t \mid q_1, \ldots, q_n)$$

Estimating relevance model (cont'd)

$$
\begin{aligned}
P(t \mid M_r) &\approx P(t \mid q_1, \dots, q_n) \\
&= \frac{P(t, q_1, \dots, q_n)}{P(q_1, \dots, q_n)} \\
&\overset{rank}{=} \sum_{d \in \mathcal{C}} P(t, q_1, \dots, q_n \mid d) P(d) \\
&= \sum_{d \in \mathcal{C}} P(d) P(t \mid M_d) \prod_{i=1}^{n} P(q_i \mid M_d) \\
&\overset{rank}{=} \sum_{d \in \mathcal{C}} w_d \cdot P(t \mid M_d), \text{ where} \\
w_d &= \prod_{i=1}^{n} P(q_i \mid M_d)
\end{aligned}
$$

$P(t \mid M_r)$ is the weighted average of $P(t \mid M_d)$ in a set of documents $\mathcal{C}$, where weights are the query likelihood scores $\prod_{i=1}^{n} P(q_i \mid M_d)$.

Relevance feedback

1. Rank results using the query likelihood score $P(q \mid d)$
2. Obtain a set of relevant results $\mathcal{C}$ through (pseudo-)relevance feedback
3. Calculate the relevance model $P(t \mid M_r)$

$$P(t \mid M_r) = \sum_{d \in \mathcal{C}} w_d \cdot P(t \mid M_d)$$

$$w_d = \prod_{i=1}^{n} P(q_i \mid M_d)$$

4. Rerank results using the negative KL-divergence score (or negative cross entropy)

$$\sum_{t \in V} P(t \mid M_r) \log P(t \mid M_d)$$

Outline

3. Language modeling in IR
   - Method
   - Relevance feedback
   - Smoothing

Jelinek-Mercer smoothing

$$P_s(t \mid M_d) = \lambda P(t \mid M_d) + (1 - \lambda) P(t \mid M_c)$$
$$= \lambda \frac{tf(t, d)}{dl(d)} + (1 - \lambda) \frac{cf(t)}{cl}$$

- $cf(t)$ – collection frequency of term t
- cl – collection length
- Smoothed query likelihood

$$P_s(q \mid M_d) = \prod_{i=1}^{n} \left[\lambda \frac{tf(q_i, d)}{dl(d)} + (1 - \lambda) \frac{cf(q_i)}{cl} \right]$$

Relationship to TF-IDF

$$\log P_s(q \mid M_d) = \sum_{i=1}^{n} \log \left[\lambda \frac{tf(q_i, d)}{dl(d)} + (1-\lambda)\frac{cf(q_i)}{cl}\right]$$

$$= \sum_{i:tf(q_i,d)>0} \log \left[\lambda \frac{tf(q_i, d)}{dl(d)} + (1-\lambda)\frac{cf(q_i)}{cl}\right]$$

$$+ \sum_{i:tf(q_i,d)=0} \log(1-\lambda)\frac{cf(q_i)}{cl}$$

$$\stackrel{rank}{=} \sum_{i:tf(q_i,d)>0} \log \frac{\lambda \frac{tf(q_i,d)}{dl(d)} + (1-\lambda)\frac{cf(q_i)}{cl}}{(1-\lambda)\frac{cf(q_i)}{cl}}$$

$$= \sum_{i:tf(q_i,d)>0} \log \left[\frac{\lambda \frac{tf(q_i,d)}{dl(d)}}{(1-\lambda)\frac{cf(q_i)}{cl}} + 1\right]$$

Dirichlet smoothing

- A unigram language model can be seen as a multinomial distribution over words $\mathcal{L}_d(n_1, \ldots, n_k \mid p_1, \ldots, p_k)$
  - $n_i = tf(t_i, d)$
  - $p_i = P(t_i \mid M_d)$
- The conjugate prior for multinomial is the Dirichlet distribution $P_{prior}(p_1, \ldots, p_k; \alpha_1^{pr}, \ldots, \alpha_k^{pr})$
  - $\alpha_i^{pr} = \mu P(t_i \mid M_c)$
  - μ is a smoothing parameter ($\lambda = \frac{dl}{dl+\mu}$)
- The posterior is the Dirichlet distribution with parameters $\alpha_i^{po} = n_i + \alpha_i^{pr} = tf(t_i, d) + \mu P(t_i \mid M_c)$
- Dirichlet smoothing

$$P_s(t \mid M_d) = \frac{tf(t_i, d) + \mu P(t_i \mid M_c)}{dl(d) + \mu}$$

Chinese restaurant process

1. Start with an empty restaurant
2. The 1st customer sits at the 1st table and chooses dish x from the restaurant's menu with probability $P(x \mid menu)$
3. The $n+1$th customer has two options
   a) Sit at the 1st unoccupied table with probability $\frac{\mu}{n+\mu}$ and choose dish x from the menu
   b) Sit at any of the occupied tables with probability $\frac{n_t}{n+\mu}$ and eat the same dish x_t as others at that table

$$P(\text{customer } n+1 \text{ eats dish } x) = \frac{\sum_{t:x} n_t + \mu P(x \mid menu)}{n+\mu}$$

Dirichlet smoothing as Chinese restaurant process

CRP	IR
dish	word
restaurant	document
menu	collection

Experimental comparison

Collection	Method	Parameter	MAP	R-Prec.	Prec@10
Trec8 T	Okapi BM25	Okapi	0.2292	0.2820	0.4380
	JM	$\lambda = 0.7$	0.2310 (p=0.8181)	0.2889 (p=0.3495)	0.4220 (p=0.3824)
	Dir	$\mu = 2,000$	**0.2470** (p=0.0757)	0.2911 (p=0.3739)	**0.4560** (p=0.3710)
	Dis	$\delta = 0.7$	0.2384 (p=0.0686)	0.2935 (p=0.0776)	0.4440 (p=0.6727)
	Two-Stage	auto	0.2406 (p=0.0650)	**0.2953** (p=0.0369)	0.4260 (p=0.4282)

Figure: TREC-8 Newswire, ad-hoc track, queries 401–450, title-only

G. Bennett, "A Comparative Study of Probabilistic and Language Models for Information Retrieval"

Experimental comparison

Collection	Method	Parameter	MAP	R-Prec.	Prec@10
TREC-2001 T	Okapi BM25	Okapi	0.1522	0.2056	0.2918
	JM	$\lambda = 0.7$	0.1113 (p=0.0003)	0.1505 (p=0.0037)	0.2122 (p=0.0003)
	Dir	$\mu = 2,000$	**0.1774** (p=0.0307)	**0.2238** (p=0.3236)	**0.3184** (p=0.3165)
	Dis	$\delta = 0.7$	0.1370 (p=0.0511)	0.1906 (p=0.053)	0.2653 (p=0.1348)
	Two-Stage	auto	0.1441 (p=0.2963)	0.1934 (p=0.3992)	0.2898 (p=0.8962)

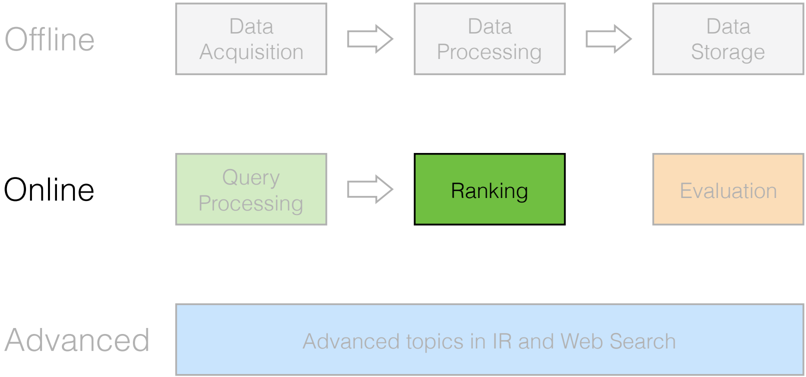
Figure: TREC-2001 Web data, ad-hoc track, queries 501–550, title-only

G. Bennett, "A Comparative Study of Probabilistic and Language Models for Information Retrieval"

Language modeling for IR summary

- Query likelihood model
- Relevance feedback
- Smoothing
  - Jelinek-Mercer smoothing
  - Dirichlet smoothing

Content-based retrieval



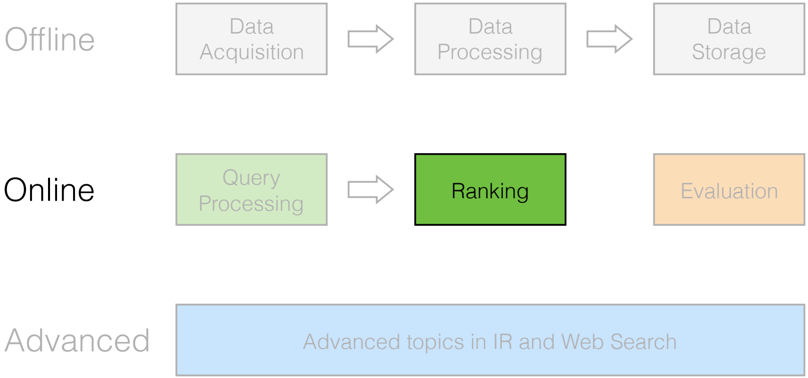
Content-based retrieval summary

- Vector space model
  - Documents and queries as vectors
  - Rank documents using cosine similarity
  - TF-IDF weights
- Probabilistic IR
  - Probability ranking principle
  - Binary independence model
  - Rank documents using odds or retrieval status value
  - BM25
- Language modeling in IR
  - Query likelihood model
  - Jelinek-Mercer and Dirichlet smoothing
- Relevance feedback

Materials

- Manning et al., Chapters 6, 9, 11, 12
- Croft et al., Chapter 7

Next lectures



Ranking methods

1. Content-based
   - Term-based
   - **Semantic**
2. Link-based (web search)
3. Learning to rank