Estimation Methods for Ranking Recent Information

Miles Efron

Gene Golovchinsky

- Presented by Avi Eini

Outline

- Introduction
- Background
- Three approaches to temporal ranking
- Experiments and conclusions

Introduction

- Recency is an important dimension of relevance for many kinds of queries. For these queries, relevant documents must not only be topically appropriate, they must also have been published in the recent past

- In the context of microblog search, fielding recency queries effectively is of prime importance

- In this paper: proposing three approaches to handling recency queries, examining their effectiveness both on microblog data, as well as on more traditional IR data in the form of two TREC news collections

Background

- Previous methods

  This paper's treatment of temporal factors in IR is based on the language modeling approach to document retrieval In particular, on the query likelihood model:

  - Query-Liklihood model:

$$Pr(d|q) \propto Pr(q|d)Pr(d)$$

assume that the prior probability distribution is over documents is uniform and that this model is a multinomial over words in the indexing language, we have the ranking function:

$$\log Pr(q|d) = \sum_{w \in q} \log Pr(w|d)$$

Background

- Previous methods

  With respect to estimating $\Pr(w|d)$ the application of a smoothing operation has been shown to play a crucial role in successful language modeling IR.

  Here is one of the simplest smoothing method:

  - Jelinek-Mercer smoothing:

$$\hat{P}r(w|d) = (1-\lambda)\hat{P}r_{ML}(w|d) + \lambda\hat{P}r(w|C)$$

$\hat{P}r_{ML}(w|d)$ - Maximum likelihood estimator ($n(w,d)/n(d)$)

$\hat{P}r(w|C)$ - Estimated probability of seeing word w in the collection

λ- Tuning parameter

Background

- Previous methods
  - Li and Croft

    proposed using the publication date of news articles to inform a prior distribution over documents rather than taking it to be uniform.

    They offered using an exponential distribution:

$$Pr(d|t_d) = r \cdot e^{-r \cdot t_d}$$

t_d - The time in months that has elapsed since document d was published

r – Rate parameter of the exponential distribution

Intuition: newer documents have higher probability than older documents do.

Background

- Previous methods
  - Dakka, Gravano and Ipeirotis

    proposed to augment the standard query likelihood framework to account for time.

$$Pr(d|q) = Pr(w_d, t_d|q) = Pr(t_d|w_d, q) \cdot Pr(w_d|q)$$

    They simply approach the problem by considering two type features for a document:

    w_d - Consists of the lexical terms in a document

    t_d - Timestamp for a document

    If we simply assume that the temporal relevance of d doesn't depend on the documents content , w, we can drop w from the joint probability:

$$Pr(d|q) \propto Pr(w_d|q) \cdot Pr(t_d|q) \propto Pr(q|w_d) \cdot Pr(w_d) \cdot Pr(t_d|q)$$

    We can see that is identical to the QL model with addition of the probability of observing time t_d given the query q

Three Approaches to temporal ranking

- Overview

  In this paper they proposed three approaches to incorporate time into retrieval paying special attention to the matter of recency queries.

  First, we introduce those approaches ,then, elaborate on each one separately:

  1. Query-specific exponential re-ranking
  2. Temporally informed smoothing
  3. Temporally biased pseudo-relevance feedback

Three Approaches to temporal ranking

- Maximum *a posteriori* estimation for exponential re-ranking

  Using an exponential distribution to accomplish a blending of time and language model has been shown to be effective in previous research. However, the typical approach to using the exponential distribution as a document prior by definition ignores query-specific concerns.

  In order to relate this issue they proposed a combination of Li and Croft and Dakka et al. approaches by letting $Pr(t_d|q)$ follow the exponential distribution.

  If the query-independent document prior is ignored, we got:

$$Pr(d|q) \propto Pr(q|d) \cdot r_q \cdot e^{-r_q \cdot t_d}$$

  r_q - Exponential rate parameter (specific for query q)

  Why is r_q important?

  - Performance of an exponential-based age penalty depends on the parameterization of the distribution
  - A rate parameter leads to improvement in recency queries appears to degrade performance on non-temporal queries

Three Approaches to temporal ranking

- Maximum *a posteriori* estimation for exponential re-ranking
  - Comparison between Exponential Prior Method (EXP) and QL

    The experiment was conducted on a Twitter data (will be discussed later), and r parameter was set to 0.015 (optimize performance of recency queries on MAP)

	Recency Queries			Non-Temporal Queries		
	MAP	**Rprec**	**NDCG**	**MAP**	**Rprec**	**NDCG**
QL	0.321	0.339	0.570	0.424	0.529	0.600
EXP	0.354+	0.373+	0.597+	0.337-	0.439-	0.519-

Conclusions:

1. As expected, for recency queries, applying an exponential prior improves retrieval on our training recency queries
2. For non-temporal queries, the exponential prior leads to a stark decrease in performance.

Three Approaches to temporal ranking

- Maximum *a posteriori* estimation for exponential re-ranking
  - How do we estimate the r_q parameter?
    1. Retrieve k documents using QL
    2. Estimate r_q based on those documents

$$\hat{r}_q^{ML} = \frac{1}{\bar{T}}$$

$T = \{t_1, t_2, \ldots, t_k\}$ - Times associated with the top k documents from the QL initial retrieval

$\bar{T}$ - Sample mean of T

When constructing r_q for recency and non-temporal queries they got:

$\hat{r}_{recency}^{ML}$ = 0.029

$\hat{r}_{non-temp}^{ML}$ = 0.017 (with P-value of 0.057)

Three Approaches to temporal ranking

- Maximum *a posteriori* estimation for exponential re-ranking
  - Problem

    In practice, MLE obtained from query information may lead to models lending to much influence to time, eclipsing lexical query similarity.

  - Solution

    Replacing the MLE with the maximum *a posteriori* estimate of r_q.

    The conjugate prior of the exponential distribution is the gamma distribution, this yields an estimation for the exponential rate parameter:

$$r_q^{MAP} = \frac{\rho + k - 1}{\sigma + \sum_{i=1}^{k} t_i}$$

    The summation over the observed document times in the denominator should reduce the temporal influence for non-temporally bound queries.

    From now on, we refer this methods as **Bayesian Exponential Ranking** (BEX)

Three Approaches to temporal ranking

- Temporally-smoothed language models

  The second approach to incorporating time into retrieval for recency queries is to smooth older documents' language models more aggressively than newer documents' models.

  Intuition:

  as documents age, we have less confidence in their precise lexical content. After many months, perhaps an author would use different words to describe the same topic.

  Formalization

  We may consider the mixing parameter λ as the probability of choosing $Pr(w|C)$ .That is, the author chooses the background model with probability otherwise favoring the MLE.

  We can relate it as a binomial distribution with λ parameter.

  Now, we can replace λ (in the Jelinek-Mercer equation) to λ_t

$$\hat{P}r(w|d) = (1 - \lambda_t)\hat{P}r_{ML}(w|d) + \lambda_t\hat{P}r(w|C)$$

  such that the older d the larger λ_t

Three Approaches to temporal ranking

- Temporally-smoothed language models

  As mentioned, we can treat λ_t as a parameter of a binomial distribution, where a corresponds to finding that *d* is old. To estimate λ_t we have the maximum likelihood estimator:

$$\hat{\lambda}_t^{MLE} = \frac{n(t<t_d)}{n(D)}$$

  $n(t<t_d)$ - number of document newer than document d

  $n(D)$ - number of documents in the collection

  As in BEX approach , they used a Bayesian approach to parameter estimation. They assume there is a prior belief about λ, specifically that λ follows a beta distribution (which is the conjugate prior to binomial) , then we have the maximum *a posteriori* estimator:

$$\hat{\lambda}_t^{MAP} = \frac{n(t<t_d)+\alpha-1}{n(D)+\beta-\alpha-2}$$

  From now on, we refer this methods as **Temporal smoothing query likelihood (TSQL)**

Three Approaches to temporal ranking

- Temporally conditioned relevance models
  - Relevance model (Laverenko and Croft)

    q - Query contains n token $q_1, q_2, \ldots, q_n,$

    $Pr(w, q_1 \ldots q_n)$ - Joint probability of word w and query terms

    k - number of top documents from an initial retrieval

$$Pr(w, q_1 \ldots q_n) = \sum_{i=1}^{k} Pr(d_i) Pr(w|d_i) \prod_{j=1}^{n} Pr(q_j|d_i)$$

This allows us to define the probability of word *w* under the relevance model *R* as:

$$Pr(w|R) = \frac{Pr(w, q_1 \ldots q_n)}{Pr(q_1 \ldots q_n)}$$

This is the basic relevance model (RM1)

Three Approaches to temporal ranking

- Temporally conditioned relevance models

  Li and Croft proposed integrating recency into relevance models. Following their use of document priors, they suggested the exponential distribution for $Pr(d_i)$.

  We refer to this approach as **exponential relevance models** (EXRM) in contrast to a baseline, non-temporal **relevance modeling approach** (RM).

  As we know, In many applications, the relevance model is interpolated with the original query model before the second retrieval via:

$$\hat{P}r(w|R) = (1-\mu)Pr(w|q) + \mu Pr(w|R) \quad \text{(RM3)}$$

  μ- A tuning parameter, controls the influence of the original query against feedback terms.

Three Approaches to temporal ranking

- Temporally conditioned relevance models

  As in TSQL model, they offered a temporal-related parameter $\hat{\mu}$ which gives more weight to feedback terms when the feedback query retrieves newer documents

  - μ parameter setting procedure

    1. Building a relevance model on q using $Pr(w|R) = \frac{Pr(w,q_1 \ldots q_n)}{Pr(q_1 \ldots q_n)}$
    2. Performing a second retrieval using KL divergence method with the relevance model
    3. Getting vector of feedback documents $t_{fb} = \{t_{fb1}, \cdots, t_{fbk}\}$
    4. Referring to μ as a binomial parameter we can use MLE estimator:

$$\hat{\mu}^{MLE} = \frac{\sum_{i=1}^{k} n(t > t_{fbi})}{k \cdot n(d)}$$

Where “success” is the observation that a particular feedback document is newer than a particular document from the collection.

Three Approaches to temporal ranking

- Temporally conditioned relevance models

  As in previous models, we can use the conjugate prior distribution (beta) to get the MAP estimator for the parameter:

$$S(\dot{t}_{fb}) = \sum_{i=1}^{k} n(t > t_{fbi})$$

$$\hat{\mu}^{MAP} = \frac{S(\dot{t}_{fb}) + \alpha - 1}{k \cdot n(D) + \beta - \alpha - 2}$$

$$\hat{Pr}(w|R) = (1 - \hat{\mu}^{MAP}) Pr(w|q) + \hat{\mu}^{MAP} Pr(w|R)$$

  From now on, we refer this methods as **Temporally smoothed relevance model (TRSRM)**

Experiments and conclusions

- Data

  Twitter data-set-

Number of Tweets	7,168,842		
Number of Tracked Users	9,274		
Earliest Date	18 Jul 2010		
Latest Date	4 Dec 2010		
Queries		Train	Test
	Recency	21	49
	Non-temp.	31	50

  - Data-set was constructed by choosing initial group of Tweeter users expended by users whom these users follow on Twitter (IR-interests group based)
  - Queries were obtained by asking two users of Twitter search engine to create queries of two types recent queries and non-temporal queries(25 for each)
  - Each query relevance judgments obtained by AMT service on a pool of 50 initial-retrieved documents (using basics models), and scored in scale from 0 to 2 with an additional “I don’t know” option, by six judges.
  - Final numeric scores were obtained by two methods:
    - Averaging – document with at least 1.5 score was counted as relevant
    - Voting – majority vote with at least three judges agreeing with same score

Experiments and conclusions

- Data

  Trec data-

	AP	LA/FT
Documents	AP (disks 1 & 2)	LA Times and Financial Times (disks 4 & 5)
Doc. Count	164,597	342,054
Dates	1 Jan '88 – 31 Dec '89	23 Apr '91 – 31 Dec '94
Topics	101-200	351-450 (test), 301-350 (train)
Recency queries	20 (test only)	24 (test), 17 (train)
Non-temp. queries	72 (test only)	65 (test), 30 (train)

- Topic was classified as "recency" if at least 2/3 form it's relevant documents appear prior to the median document time
- Only queries with at least 10 relevant documents was counted.

Experiments and conclusions

- Parametrization
  - Because optimal values were very close for the TREC and Twitter data a single parametrization has been chosen
  - To make time-smoothed comparable to standard language models smoothed with $\lambda = 0.4$, they choose α and β such that:

    $$(\alpha - 1)/(\beta - \alpha - 2) = 0.4$$

    leaving only the magnitude of β as tunable parameter
  - A standard stop-list was applied

Param	Description	Eqs.	Value
n	Num. Docs retrieved per query	NA	100
k	Num. top docs. Used for calculations	9, 13	20
r	Rate parameter for exponential priors	4	0.015 (TREC), 0.01 (Twitter)
λ	Parameter for Jelinek-Mercer smoothing	3	0.4
ρ	Effective sample size for exponential MAP estimate (BEX)	9	100
σ	Shape parameter for exponential MAP estimate (BEX)	9	Derived
β	Effective sample size for binomial MAP estimate (TSQL, TSRM)	10, 17	2*$n(D)$ (TSQL) 250 (TSRM)
α	Hyperparameter for binomial MAP estimate (TSQL, TSRM)	10, 17	derived2
μ	Mixing parameter for relevance models (RM, EXRM)	15	0.4
f	Number of expansion terms used in pseudo-RF	NA	10

Experiments and conclusions

- Experimental Results
  - Baselines and models

    QL and RM are the baselines for term-based methods and feedback runs methods respectively (noted in grey)

Abbreviation	Description
QL	Query likelihood (non-temporal)
EXP	Exponential priors as in [15]
BEX	Bayesian exponential ranking
TSQL	Time-smoothed query likelihood
RM	Relevance models (non-temporal)
EXRM	Exponential relevance models as in [15]
TSQL	Time-smoothed relevance models

Experiments and conclusions

- Experimental Results
  - results

Retrieval effectiveness on TREC AP Data

	Recency Queries			Non-Temporal Queries		
	MAP	**Rprec**	**NDCG**	**MAP**	**Rprec**	**NDCG**
QL	0.198	0.259	0.401	0.150	0.213	0.302
EXP	0.204	0.265	0.408	0.145-	0.208	0.296-
BEX	0.205+	0.265	0.408+	0.147	0.209	0.300
TSQL	0.203+	0.263	0.405	0.148	0.210	0.302
RM	0.267	0.318	0.469	0.192	0.262	0.357
EXRM	0.266	0.317	0.463	0.191	0.262	0.357
TSRM	0.272+	0.323+	0.473+	0.192	0.263+	0.358

Retrieval effectiveness on TREC LA/FT Data

	Recency Queries			Non-Temporal Queries		
	MAP	**Rprec**	**NDCG**	**MAP**	**Rprec**	**NDCG**
QL	0.175	0.225	0.324	0.142	0.208	0.306
EXP	0.187+	0.238	0.332	0.134-	0.198-	0.209-
BEX	0.186+	0.241+	0.331+	0.138	0.201-	0.297-
TSQL	0.184+	0.241+	0.326	0.138	0.199-	0.300
RM	0.193	0.244	0.335	0.152	0.222	0.319
EXRM	0.197	0.248	0.340	0.151	0.219-	0.318
TSRM	0.196+	0.244	0.339+	0.152	0.219-	0.318

Experiments and conclusions

- Experimental Results
  - Results cont.

Retrieval Effectiveness on Twitter Data. Relevance judgments based on averaging and voting methods

	Relevance by Averaging						Relevance by Voting					
	Recency Queries			Non-Temporal Queries			Recency Queries			Non-Temporal Queries		
	MAP	Rprec	NDCG	MAP	Rprec	NDCG	MAP	Rprec	NDCG	MAP	Rprec	NDCG
QL	0.340	0.409	0.576	0.336	0.358	0.535	0.325	0.390	0.573	0.276	0.285	0.488
EXP	0.364+	0.420	0.596	0.305-	0.338	0.486-	0.343+	0.411+	0.589	0.244-	0.267-	0.430-
BEX	0.362+	0.421	0.597+	0.317-	0.349	0.499-	0.344+	0.406+	0.591	0.256-	0.274	0.444-
TSQL	0.361+	0.418	0.594	0.335	0.360	0.528	0.347+	0.408+	0.589	0.275	0.287	0.478
RM	0.313	0.399	0.549	0.332	0.367	0.517	0.312	0.383	0.541	0.279	0.298	0.477
EXRM	0.313	0.394	0.538	0.339	0.360	0.526	0.304	0.377	0.534	0.277	0.302	0.477
TSRM	0.315	0.390	0.537	0.337	0.359	0.525	0.306	0.384	0.532	0.277	0.299	0.475

Experiments and conclusions

- Experimental Results

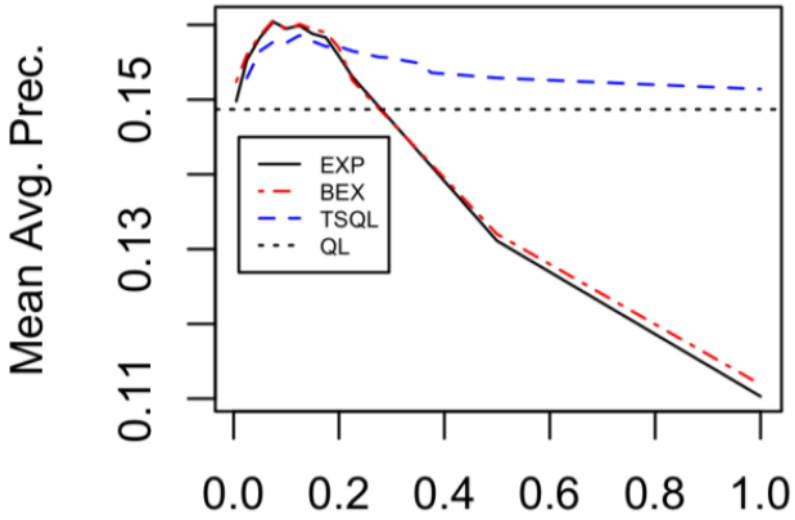
  Some conclusions:

  1. For recency queries, all three non-feedback models improved MAP for all datasets significantly
  2. The difference in effectiveness (on any of our three measures) between EXP and BEX or TSQL was never observed to be statistically significant on the recency queries
  3. The method based on exponential document priors saw a statistically significant decline in MAP in comparison with the non-temporal QL for all tested non-temporal queries
  4. BEX *did* show declines in performance for recency queries compared to QL. But these declines were less severe than in the case of EXP
  5. TSQL saw a statistically significant decline in performance only once in our experiments
  6. relevance feedback hindered retrieval on the Twitter data, with MAP significantly lower for RM than for QL in all cases but one

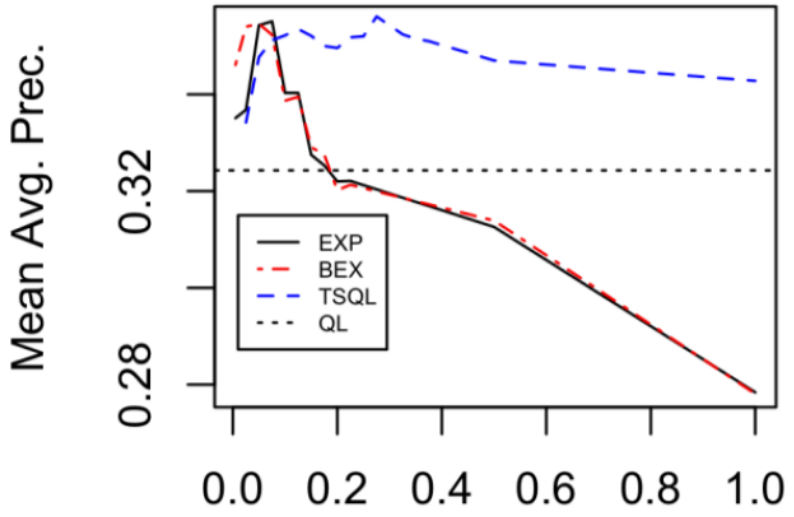
Experiments and conclusions

- Sensitivity of Models to Parameterizations

Effect of model parametrization on MAP for TREC training data



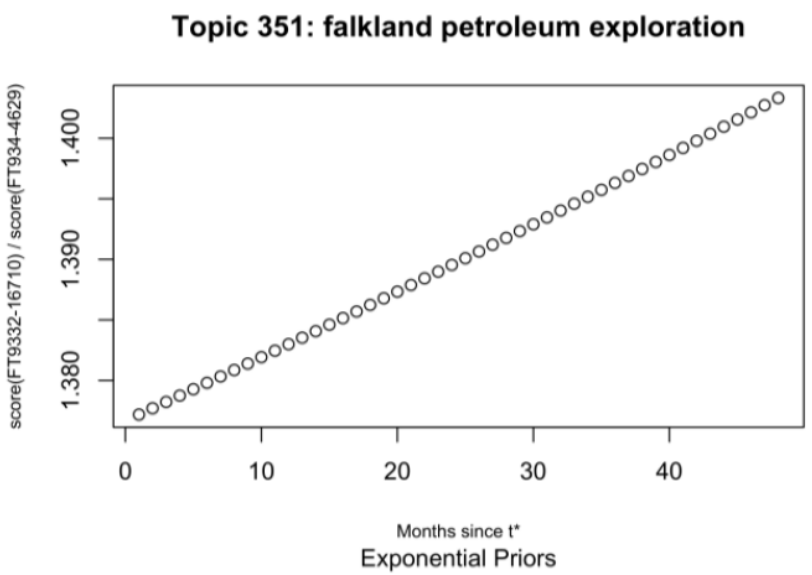
Effect of model parametrization on MAP for Twitter data

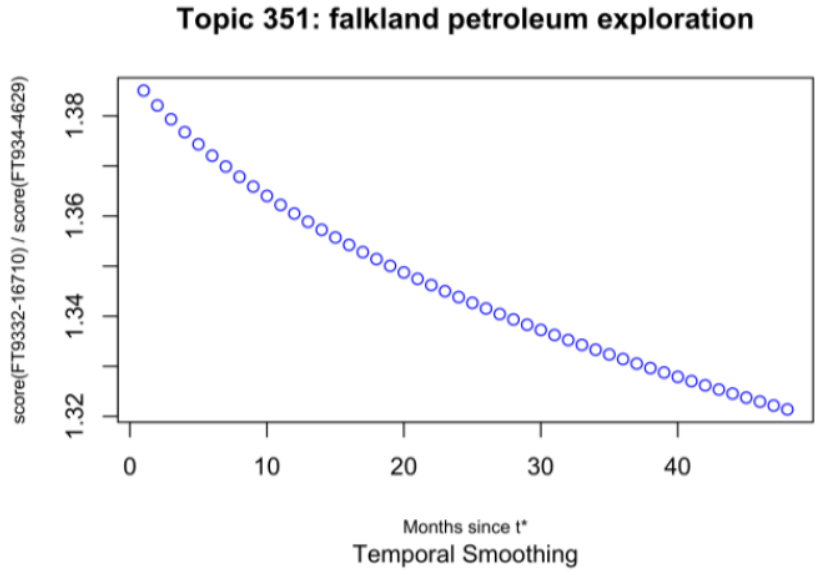


- We can notice that in the right panel the influence of parametrization over the TSQL model is negligible (due to the setting of the hyperparameters to a proportion of 0.4)

Experiments and conclusions

- Performance of Time-Smoothed Language models





	petroleum	exploration	falkland	$n(d)$
FT932-16710	2	1	5	292
FT934-4629	1	2	0	79

- FT932-16710 has higher likelihood than FT934-4629 using a constant $\lambda = 0.4$
- By altering the timestamp of the documents making them appear older, we can see major differences in the graphs. While in EXP the documents, increasingly become less similar, in TSQL the similarity increase as they get older

Thanks