Learning to Rank for Freshness and Relevance

Na Dai Milad Shokouhi Brian D.Davison

- Presented by Avi Eini

Outline

- Introduction to Temporal IR and it’s challenges
- Background
- CS-DAC (Criteria-Sensitive Ranking Divide And Conquer)
- Experiments and conclusion

Introduction

- What is temporal IR?

    - A sub-field of IR focusing in retrieving search result for a queries with a temporal characteristics .
    - For example: News search , Ipad , “US open”

- What are the challenges temporal IR dealing with?

    - Freshness VS Relevance
    - Temporal Features

Freshness vs Relevance

- Relevance
  - Quantify the topical matchability between query and web-pages or documents
  - Well-studied measure

- Freshness
  - an actual page contents reflects new information (for which query?)
  - Recency of page maintenance (with respect to the time point of generating ranking lists)

- Are the above two definitions of Freshness are the same?

- The challenge
  - Acquiring a LTR ranker consider both Freshness and Relevance as an optimization goal

Background

- Pairwise learning to rank
  - Given a query and a pair of associated documents, if one is more relevant than other, then it is boosted in the training process to get higher rank
  - Query-type information ignored
- Query-dependent loss/ranking
  - Query-dependent ranking (K-Nearest Neighbor , Geng et al.)
  - Query-dependent loss- learning multiple ranking functions and query categorization (Bian et al.)
  - Problems:
    - Query categorization may not be available
    - Categorization may not be fine-grained

Background

- DAC – Divide And Conquer for ranking

  The DAC ranking framework can be summarized in three main steps:

  1.Identifying ranking-sensitive query categories

  queries are categorized according to their ranking characteristics and ranking features discriminativity (Divide step)

  2.Learning topic-specified ranking models via minimizing a global risk

  Training of multiple ranking models (one for each cluster) by a unified learning method, while each query contribute to a specific ranker according to it's belonging to the cluster

  3.Running each unseen query against all ranking models

  At last, each unseen query is submitted to all ranking models and a weighted combination is used to merge the final results

Background

- Multiple Criteria Ranking

  Training ranking models for multiple criteria beyond relevance, such as diversity efficiency

  Dong et al.'s work

  - Consider Freshness in instance labeling for training effective learning models.
  - Demote the relevance labels of stale pages
  - Show significant improvements in both Relevance and Freshness

  In this paper

  - Generating *hybrid* labels for documents based on their Relevance and Freshness
  - Multi-Criteria ranking model in DAC framework

Background

- Temporal signals for ranking

  temporal signals captures the dynamics of the queries ,web-pages ,hyperlinks in order to improve search quality

  Typically, temporal signals are complementing the content-based matching scheme

  - Profiling query temporal characteristics
  - Emphasizing documents whose temporal characteristics are close to the query's temporal profile

  Other works incorporated the dynamics of the content changes into document language model

Criteria Sensitive Ranking

- CS-DAC

  A ranking framework incorporates the balance between relevance and freshness into training customized rankers that optimize both freshness and relevance

  A typical ranking function f with ω parameters takes a query-document feature vector X as input and produces ranking scores of documents.

$$\hat{\mathbf{y}} = f(\mathbf{X}, \omega)$$

  The goal is to find a ranking model f^* that that takes query-document feature vectors as input ,and produces a document ranking as accurate as possible

$$f^* = \arg\min_f \sum_q \mathcal{L}(f(\mathbf{X}_q, \omega), \mathbf{y}_q) = \arg\min_f \sum_q \mathcal{L}(\hat{\mathbf{y}}_q, \mathbf{y}_q)$$

Criteria Sensitive Ranking

- CS-DAC Framework

  As mentioned before, in DAC framework we cluster queries based on their feature representation and separate rankers are trained with each cluster simultaneously

  Each ranker $f_i{}^*$ is learned via:

$$f_i^* = \arg\min_{f_i} \sum_{q \in \mathcal{Q}} \mathcal{I}(q,i)\mathcal{L}_i(\hat{\mathbf{y}}_q, \mathbf{y}_q)$$

$\mathcal{Q}$ - Training query set

$\mathcal{I}(q,i)$ - Importance of query q with respect to the i^{th} ranking model

$\mathcal{L}(\hat{\mathbf{y}}_q, \mathbf{y}_q)$ - Loss function

Criteria Sensitive Ranking

To account for relevance and freshness simultaneously, they propose to use hybrid labels that are generated based on freshness and relevance judgments

The hybrid labels is based on a weighted harmonic mean function which maps relevance and freshness grades to a single equivalent numerical score $\widetilde{y}_{q,d}$

$$\widetilde{y}_{q,d,i} = \frac{(1+\beta_i^2) \cdot y_{q,d}^R \cdot y_{q,d}^F}{y_{q,d}^R + \beta_i^2 \cdot y_{q,d}^F}$$

$y_{q,d}^R$ -Relevance grade on the query document pair $<q,d>$

$y_{q,d}^F$ -Freshness grade on the query document pair $<q,d>$

β_i -Parameter for setting the trade-off between relevance and freshness

Criteria Sensitive Ranking

Allowing different values of β for rankers means that each query-document pair may affect the pairwise learning of each ranker differently Therefore they suggest to factorize query-document pair importance as follows

$$f_i^* = \arg\min_{f_i} \sum_{q \in \mathcal{Q}} \mathcal{I}(q,i) \times \sum_{<d_1,d_2> \in \mathcal{D}_q} \mathcal{U}'(q,i,d_1,d_2) \mathcal{L}_i\left(\begin{bmatrix} \hat{y}_{q,d_1,i} \\ \hat{y}_{q,d_2,i} \end{bmatrix}, \begin{bmatrix} \widetilde{y}_{q,d_1,i} \\ \widetilde{y}_{q,d_2,i} \end{bmatrix} \right)$$

$\mathcal{D}_q$ - set of preferential query-documents pair with respect to query q

$\mathcal{U}'(q,i,d_1,d_2)$ - importance of $<d_1, d_2>$ in training for query q with respect to the i^{th} ranking model

Criteria Sensitive Ranking

For simplicity, we assume $<q, d_1>$ and $<q, d_2>$ are independent, and so factorize the importance of the preferential pair $\mathcal{U}'(q, i, d_1, d_2)$ as follows.

$$\mathcal{U}'(q, i, d_1, d_2) = \mathcal{U}(q, i, d_1) \cdot \mathcal{U}(q, i, d_2)$$

where $\mathcal{U}(q, i, d_1)$ is the importance of query-document pair $<q, d_1>$ in training for query q with respect to the i^{th} ranking model

- Is the Independence assumption holds?

Criteria Sensitive Ranking

- Ensemble ranking

  Given an unseen query q_0:

  1. profile its query characteristics

  2. calculate its distances to the centroids of existing query clusters $c_1, c_2, \dots c_n$.

  3. score the trained ranking functions according to the normalized distance between the query and their corresponding clusters, given by:

$$W_i = \frac{\mathcal{I}(q', i)}{\sum_{i'=1}^{n} \mathcal{I}(q', i')}$$

  4. The query q' is run against all n rankers and the final result produced according to the ensemble ranking of their outputs:

$$\theta_{q'} = \sum_{i=1}^{n} W_i f_i^* (\mathbf{X}_{q'}, \omega_i)$$

Criteria Sensitive Ranking

- Query importance $(\mathcal{I})$

  The $\mathcal{I}(q,i)$ values provide a Binomial distribution over each of criteria-sensitive query clusters, and specify the importance of different ranking functions. Gaussian Mixture model (GMM) was used as soft k-means clustering to group queries into clusters.

  The importance of a query to cluster is given by:

$$\mathcal{I}(q,i) = 1 - \frac{\|\mathbf{p}_q - \mathbf{c}_i\|^2}{\max_{q' \in \mathcal{Q}} \|\mathbf{p}_{q'} - \mathbf{c}_i\|^2}$$

  $\boldsymbol{p}_q$ - Feature vector of query

  $\boldsymbol{c}_i$ - Centroid of the cluster

Criteria Sensitive Ranking

- Document importance $(\mathcal{U})$
  - In pairwise learning to rank methods, the importance of a document with label y during training depends on the number of times it is compared to other documents with different labels
  - In our model, because of the parameter β ,we might have unequal labels coming from the same initial label for different rankers $\longrightarrow$ **problem**

To over this problem, they suggest a factorization component $\mathcal{U}$ which estimate the importance of a query-document pair by the likelihood of visiting that label

$$\mathcal{U}(q,i,d) = \frac{\sum_{q' \in \mathcal{Q}} N(q', i, \mathbf{y}_{q,d}) \cdot N(q', i, \neg \mathbf{y}_{q,d})}{\sum_{\mathbf{y}' \in \mathcal{Y}_i} \sum_{q' \in \mathcal{Q}} N(q', i, \mathbf{y}') \cdot N(q, i, \neg \mathbf{y}')}$$

$\mathcal{Y}_i$ - Space of labels for ranker q
$N(q,i,\mathbf{y})$ - number of documents with label **y**
$N(q,i,\neg\mathbf{y})$ - number of documents without label **y**

Criteria Sensitive Ranking

- Document importance $(\mathcal{U})$
  - Problems:
    1. additional inter-label dependencies may arise from comparing common labels
    2. Overemphasizing certain documents inevitably introduces bias in ranking

  - Solution: smoothing documents importance values by using Random Walk approach

    1.Define $\mathcal{U}(q, i, d)$ as $w(\mathbf{y}_{q,d})$

    2.Construct a fully-connected $w(\mathbf{y})$ bipartite graph $G(V, E)$ which each node v is a unique hybrid label $\boldsymbol{y}$ associated with $w(\mathbf{y})$ and e is associated with the number of times the labels were compared during training.

    3.Perform a random walk where at each step, the random walk surfer jumps to random node with probability d or follows some connected

Criteria Sensitive Ranking

- Loss function $(\mathcal{L})$
  - RankSVM:

$$\arg\min_{\omega,\xi_{q,i,j}} \frac{1}{2}\|\omega\|^2 + C\sum_{q,i,j}\xi_{q,i,j} \quad \textit{subject to} \quad \begin{aligned} \forall y_i^q \succeq y_j^q :&\quad \omega^T X_i^q \geq \omega^T X_j^q + 1 - \xi_{q,i,j}, \\ \forall_q \forall_i \forall_j :&\quad \xi_{q,i,j} \geq 0 \end{aligned}$$

$\xi_{q,i,j}$ - Non-negative slack variable

C - Parameter for setting trade-off between training error and margin size

X_i^q - Query-document feature-vector

$y_i^q \succeq y_j^q$ - notation for implying document i is ranked higher than document j with respect to query q

  - Modified RankSVM (with respect to CS-DAC):

$$\arg\min_{\omega_i,\xi_{q,j,k}} \frac{1}{2}\|\omega_i\|^2 + C\sum_{q,j,k}\xi_{q,j,k} \quad \textit{subject to,} \quad \begin{aligned} \forall \widetilde{y}_{q,j,i} \succeq \widetilde{y}_{q,k,i} :&\ \mathcal{I}(q,i)\mathcal{U}(q,i,j)\omega_i^T X_j^q \\ &\geq \mathcal{I}(q,i)\mathcal{U}(q,i,k)\omega_i^T X_k^q + 1 - \xi_{q,j,k}, \\ \forall_q \forall_i \forall_j :&\quad \xi_{q,i,j} \geq 0 \end{aligned}$$

Criteria Sensitive Ranking

- Evaluation metric

  - Hybrid NDCG

    hybrid NDCG extends the commonly used evaluation metric NDCG to take hybrid labels for evaluation.

    Formally, hybrid NDCG defined as below:

$$hybrid\ NDCG(n) = Z_n \sum_{j=1}^{n} \frac{2^{(\gamma \mathbf{y}_R + (1-\gamma)\mathbf{y}_F)} - 1}{\log_2(j+1)}$$

Z_n - Oracle discounted cumulative gain in cut-off n

$\mathbf{y}_R$ - Relevance label

$\mathbf{y}_F$ - Freshness label

γ - Parameter specifying the trade-off between Freshness and Relevance

Experimental setup

- Data
  - 158 million unique URLs and 12 billion links , January 2000 to December 2007
  - One snapshot per month (88 totally)
  - Remove pages with less than 5 snapshots and kept 3.8 million pages and 435 million links

- Queries
  - 90 temporal queries sampled from “Google Trends”
  - 90 non-temporal queries sampled from MSN query log

- Judgments and Metrices
  - Freshness and Relevance was judged in scale of 0-4 separately
  - Freshness and Relevance were evaluated by hybrid NDCG

Experimental setup

- Ranking features
  - Non-temporal features

    include several commonly used text-similarity scores such as BM25 and language modeling computed over different fields of documents, and link-based static features

Table 1: Non-temporal ranking features used by RankSVM in the CS-DAC framework and baseline methods. Body, title, heading and anchor-text fields are respectively represented by B, T, H and A.

Feature name	Feature description	Feature name	Feature Description
Okapi(B)	Okapi BM25 score [25] for body-text.	RQT(B)	Ratio of covered terms in body-text.
RQT(H)	Ratio of covered terms in heading-text.	LM.JM(B)	body-text language modeling (Jelinek-Mercer) score [31].
LM.Dir(B)	Body-text language modeling (Dirichlet) score [31].	RQT(T)	Ratio of covered terms in title-text.
InNum	Number of inlinks.	TF(B)	Term frequency in body-text.
AvgNTF(B)	Average normalized TF in body-text.	LM.JM(T)	title-text language modeling (Jelinek-Mercer) score
STFIDF(H)	Sum of term TFIDF in heading-text.	NumQT(A)	Number of covered terms in anchor-text.
MaxNTF(B)	Maximum normalized TF in body-text.	PR	PageRank score [4].
AvgNTF(T)	Avgerage normalized TF for title-text.	LM.Dir(T)	title-text language modeling (Dirichlet) score.
MxTFIDF(T)	Maximum term TFIDF in title-text.	MaxNTF(T)	Maximum normalized TF in title-text.
LM.Dir(H)	heading-text language modeling (Dirichlet) score	MaxTF(T)	Maximum query term frequency in title-text.
ATFIDF(T)	Average term TFIDF in title-text.	AvgTF(T)	Average query term frequency in title-text.
SumTF(T)	Sum of term frequency in title-text.	LM.JM(H)	heading-text language modeling (Jelinek-Mercer) score.
L(B)	Body-text length.	AvgTF(H)	Average query term frequency in heading-text.
SumTF(H)	Sum of term frequency in heading-text.		

Experimental setup

- Ranking features
  - Temporal features
    - Measuring the changes in content documents (e.g. TF-IDF) and construct a time-series for different document labels (body ,heading, etc.)
    - Use STL (*seasonal-trend decomposition)* to decompose each time-series into three components:

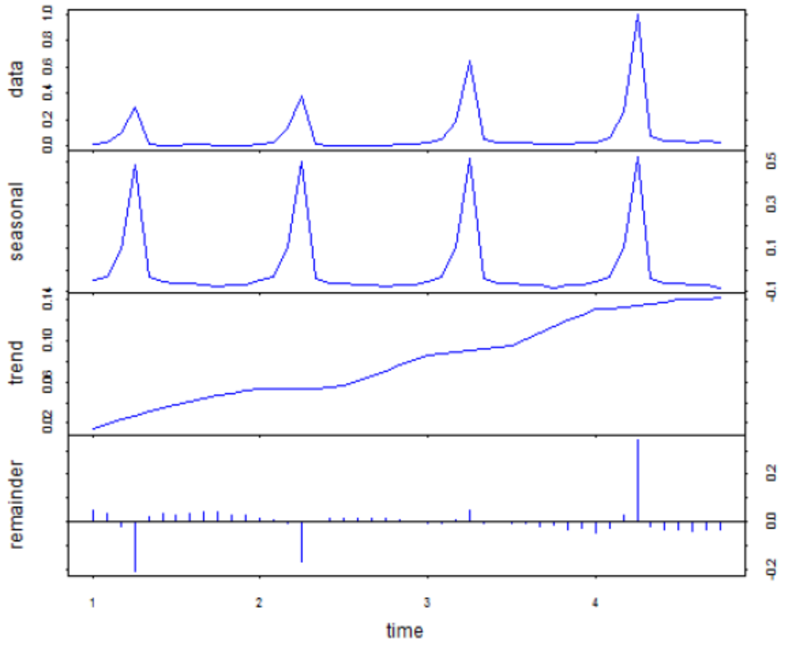
$$STL(\tau) = \mathcal{T}_\tau + \mathcal{S}_\tau + \mathcal{R}_\tau$$

$(\mathcal{T})$-Trend

$(\mathcal{S})$ -Seasonal

$(\mathcal{R})$ -Reminder

STL decomposition of click histogram



Experimental setup

- Ranking features
  - Temporal features

Feature name	Feature description
Slp(τ)	Slope of trend component T_τ.
Amp(τ)	Amplitude of seasonal component S_τ.
Rp(τ)	Relative position in S_τ.
Cs(τ)	Confidence of seasonality.
Cr(τ)	Confidence of regularity.
TPR	Timed PageRank [30].

  - The slope of the trend component captures the speed of the content changes
  - The amplitude of seasonal component captures the scale of the content changes

Experimental setup

- Query clustering features
  - The query importance features $(\mathcal{I})$ are used to cluster queries and assign the weights in each corresponding ranking function
    1. First, a reference ranker (BM25) used to make an initial retrieval
    2. The k first document (15) are gathered
    3. Average value of each Ranking feature is computed and the final mean value is the clustering feature
    4. The feature importance is computed by training a reference RankSVM model for hybrid NDCG on the training dataset

Experimental setup

- Baseline methods
  - SinR- single RankSVM ranker with all features
  - SepR – separate RankSVM rankers for temporal and non-temporal queries (assuming that the type of the query is well-known)
  - Over-Weighting model –combines Relevance and Freshness labeled data to a single ranker and over-weighting training pairs of the criterion with fewer labels.

$$\arg\min_{\omega,\xi_{q,i,j}} \frac{1}{2}\|\omega\|^2 + C\sum_{q,i,j}\xi_{q,i,j} \quad \textit{subject to} \quad \forall y_i^q \succeq y_j^q : \begin{cases} \frac{\alpha}{N_T}\omega^T X_i^q \geq \frac{\alpha}{N_T}\omega^T X_j^q + 1 - \xi_{q,i,j} & q \in \mathcal{Q}_T \\ \frac{1-\alpha}{N_N}\omega^T X_i^q \geq \frac{1-\alpha}{N_N}\omega^T X_j^q + 1 - \xi_{q,i,j} & q \in \mathcal{Q}_N \end{cases}$$

$$\forall_q \forall_i \forall_j : \quad \xi_{q,i,j} \geq 0$$

  - TopicalSVM - trains all rankers using global loss function but doesn't factorize the query-document importance (as CS-DAC)

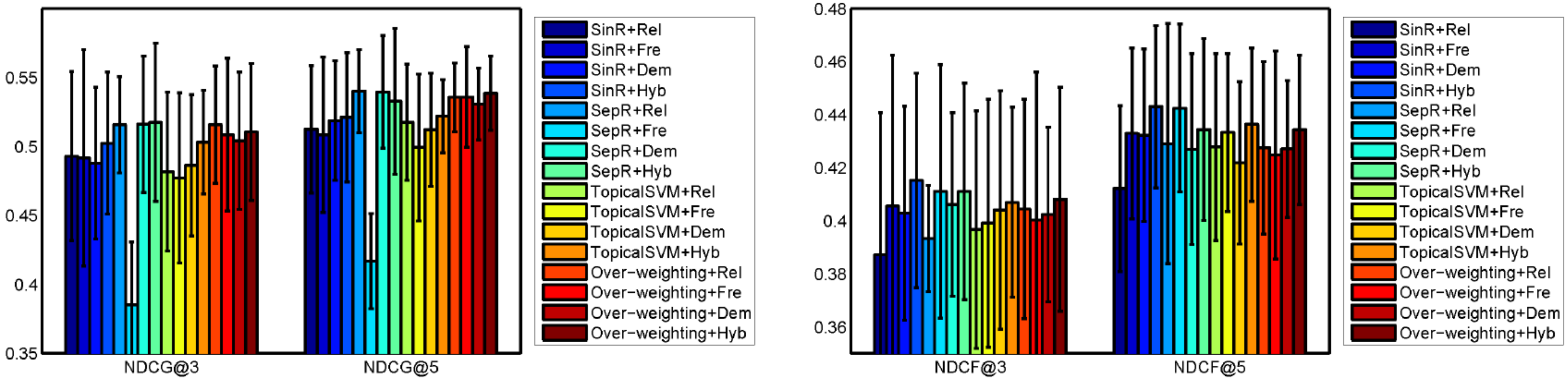
Experiments

- Baseline investigation
  - Run baselines techniques optimized for different goals
  - pick the best preforming baselines and compare the against CS-DAC

- Performance analysis for experimental baslines
  - The baselines methods were trained for one of four optimization goals:
    1. Relevance- using Relevance label only
    2. Freshness – using Freshness label only
    3. Hybrid – using hybrid labels
    4. Demoted – rescored Relevance labels according to their outdate time

Experiments

Non-temporal queries



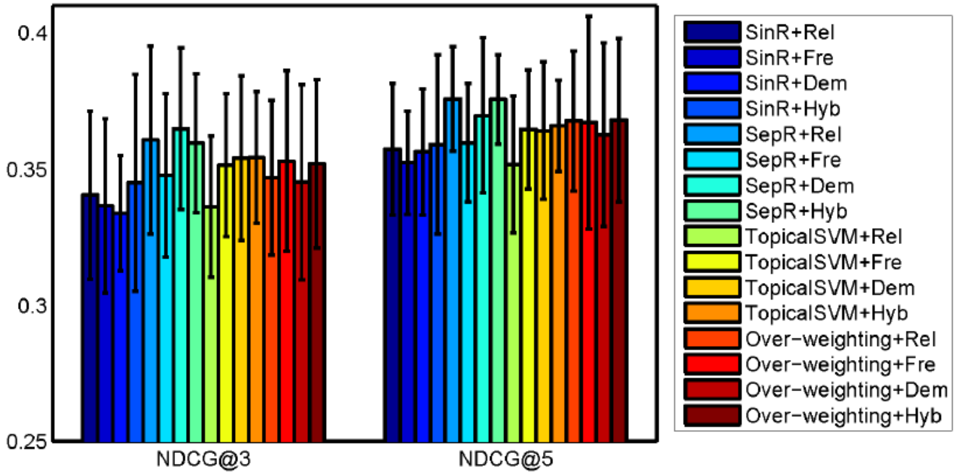
(a) *Relevance labels* $(\mathbf{y}^R)$

(b) *Freshness labels* $(\mathbf{y}^F)$

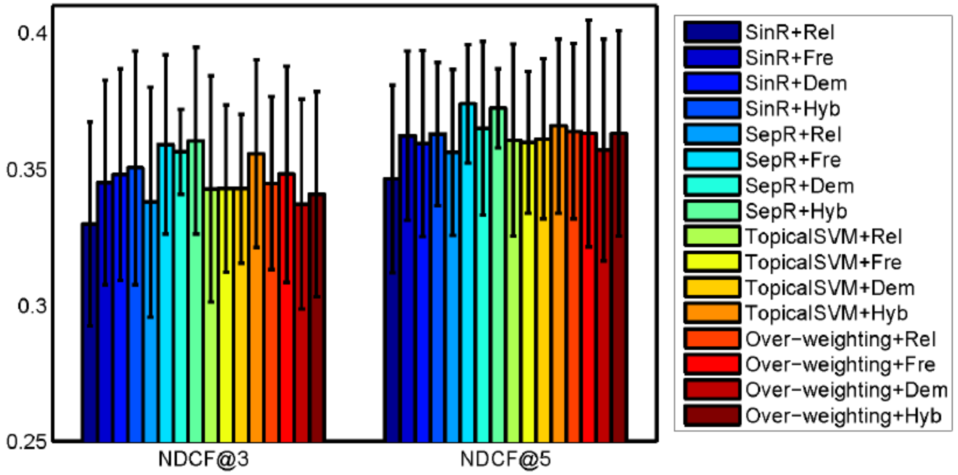
- Consequences
  1. When evaluating on Relevance labels it is more effective to optimize over for Relevance
  2. The methods optimized fir demoted and hybrid label consistently outperform over others
  3. SinR has overall poorest performance, where the other methods show similar effectiveness

Experiments

Temporal queries



(a) *Relevance labels* ($\mathbf{y}^R$)



(b) *Freshness labels* ($\mathbf{y}^F$)

- Consequences
  1. Less variation in performance compared to the non-temporal queries
  2. SinR has overall poorest performance, where the other methods show similar effectiveness

Experiments

- As a result from the baselines investigation, was performed another experiment know comparing SepR, TopicalSVM AND Over-weighted methods to CS-DAC with and without factor $\mathcal{U}$

Comparative performance on Freshness

	Temporal Queries (Google Trends)			
	NDCF1	NDCF3	NDCF5	NDCF10
SepR	0.378	0.360	0.372	0.408
TopicalSVM	0.365	0.355	0.365	0.402
Over-weighting	0.340	0.348	0.363	0.404
CS-DAC	0.398‡	0.364	0.376	0.411
CS-DAC($\mathcal{U}$)	0.416†§‡	0.379‡	0.388	0.400
	Non-Temporal Queries (MSN logs)			
	NDCF1	NDCF3	NDCF5	NDCF10
SepR	0.348	0.411	0.434	0.475
TopicalSVM	0.355	0.408	0.430	0.485
Over-weighting	0.335	0.408	0.434	0.480
CS-DAC	0.427†§‡	0.454†§‡	0.473†§‡	0.510§‡
CS-DAC($\mathcal{U}$)	0.452†§‡	0.466†§‡	0.488†§‡	0.527†§‡

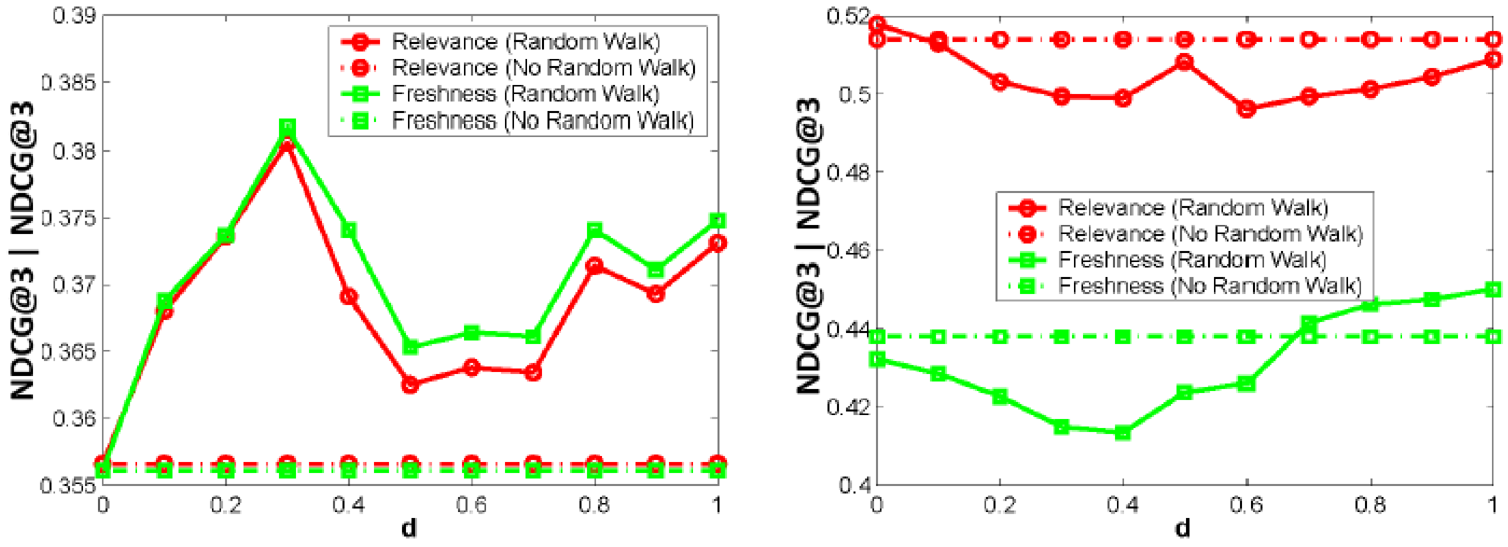
Comparative performance on Relevance

	Temporal Queries (Google Trends)			
	NDCG1	NDCG3	NDCG5	NDCG10
SepR	0.373	0.359	0.375	0.411
TopicalSVM	0.342	0.354	0.365	0.408
Over-weighting	0.355	0.351	0.368	0.411
CS-DAC	0.385	0.365	0.377	0.417
CS-DAC($\mathcal{U}$)	0.401†‡	0.375	0.389	0.426†
	Non-Temporal Queries (MSN logs)			
	NDCG1	NDCG3	NDCG5	NDCG10
SepR	0.481	0.517	0.532	0.562
TopicalSVM	0.490	0.508	0.521	0.566
Over-weighting	0.476	0.510	0.538	0.570
CS-DAC	0.493	0.520	0.541	0.574
CS-DAC($\mathcal{U}$)	0.509	0.522	0.541	0.574

Parameter analysis

- Smoothing query-document importance

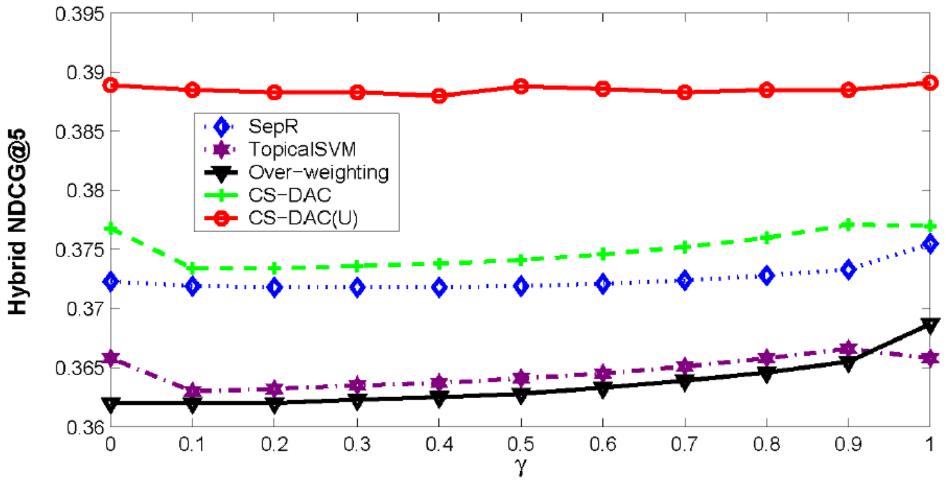
  original query-document importance values can be smoothed by random walk, where the probability d of random jumping can be tuned during training and validation.



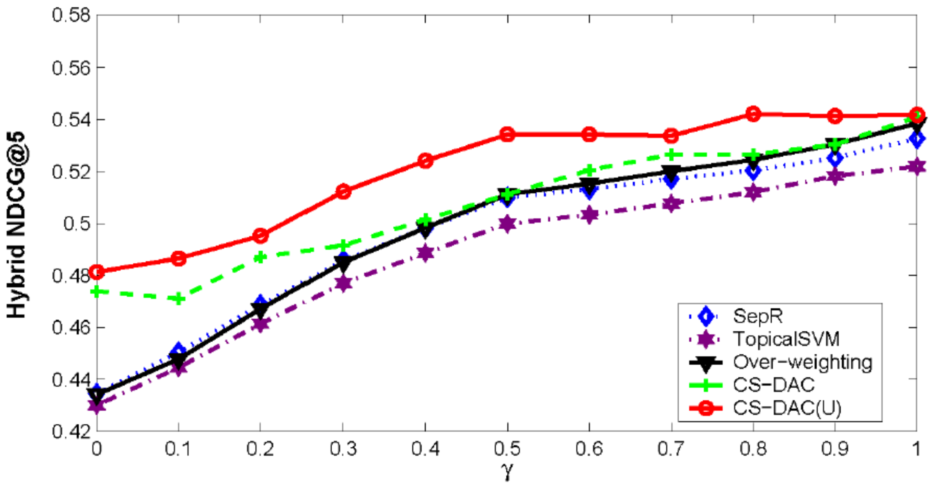
(a) *Temporal queries* (b) *Non-temporal queries*

Parameter analysis

- Hybrid labels for evaluation



(a) *Temporal queries*



(b) *Non-temporal queries*

- The (almost) monotonic grow in the non-temporal queries graph may caused by the fact that generally relevance based NDCG values are greater than those computed on the freshness labels

Parameter analysis

- Feature analysis

  - Temporal features

    - The confidence value for the seasonality and regularity of STL decomposition were generally more effective.
    - Features that was generated out the time-series decomposition of changes in the anchor-text and inlinks were more successful than those similarly produced based on other fields

  - Non-temporal features

    - BM25 and language modeling scores had the highest weights and were most effective when computed over the body and title text