Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Highlights

Field features: The impact in learning to rank approaches

Applied Soft Computing xxx (xxxx) xxx

Hua Yang*, Teresa Gonçalves*

- Field grouped features are more effective than a naively combined feature list in learning-to-rank approaches.
- The contributions of different fields are analyzed across benchmark datasets.
- The aggregation results of different fields are effective in field learning-to-rank approaches.
- The correlations between features of different fields and their combinations are discussed.

Graphical abstract and Research highlights will be displayed in online search result lists, the online contents list and the online article, but will not appear in the article PDF file or print unless it is mentioned in the journal specific style requirement. They are displayed in the proof pdf for review purpose only.

Please cite this article as: H. Yang and T. Gonçalves, Field features: The impact in learning to rank approaches, Applied Soft Computing (2023) 110183, https://doi.org/10.1016/j.asoc.2023.110183.

ASOC: 110183

ARTICLE IN PRESS

Applied Soft Computing xxx (xxxx) xxx

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Conference

Field features: The impact in learning to rank approaches

Hua Yang [a,b,*], Teresa Gonçalves [b,c,*]

[a] School of Computer Science, Zhongyuan University of Technology, Zhengzhou 450007, China
[b] Department of Informatics, Universidade de Évora, Portugal
[c] Centro ALGORITMI, Vista Lab, Universidade de Évora, Portugal

ARTICLE INFO

Article history:
Received 3 August 2022
Received in revised form 18 January 2023
Accepted 27 February 2023
Available online xxxx

Keywords:
Learning to Rank
Field Features
Aggregation
Information Retrieval

ABSTRACT

Learning to Rank approaches employ Machine Learning techniques for Information Retrieval. Traditionally, the features needed to train a ranking model are naively combined after being extracted from the various fields of the texts. Nevertheless, if not considered carefully, the learning process can make use of strongly correlated features. Moreover, the learned ranking models are not, to date, systematically analyzed in terms of how the field-based features affect their performances. In this work, the impact of using field-based features in Learning to Rank approaches is investigated. Specifically, the Field Learning to Rank technique is proposed to study if the field-based features perform better than the naively combined features. The experiments are conducted employing eight learning to rank approaches on two sizable benchmark datasets: MQ2007 and MQ2008. The models are assessed using three widely adopted Learning to Rank evaluation measures, namely Precision, Mean Average Precision, and Normalized Discounted Cumulative Gain. The results show that the use of field-based features achieve better performance than the naively combined features. Moreover, models aggregated from different fields further improve the ranking results. It is also observed that among the five examined fields, *url* and *title* are significantly more effective than *wholedoc* (full document), *body*, and *anchor* to build ranking models. Further, analyses indicate the existence of strong correlations between field features, such as the features from *body* and *wholedoc*, *title* and *anchor*, or *title* and *url*. The proposed Field Learning to Rank technique is shown to have the advantage of avoiding the combination of correlated features. These findings imply that the use of field-based features for training ranking models is valuable for enhancing the effectiveness of Learning to Rank approaches.

1. Introduction

In recent years, Learning to Rank (LTR) has been an interesting research topic in Information Retrieval (IR) and Machine Learning (ML). More specifically, LTR employs ML techniques to build ranking models for IR systems. This paper investigates the impact of field features for LTR approaches.

In state-of-the-art LTR techniques, features are usually derived from various document fields, such as *anchor*, *title*, or *url* [1–5]. The field itself can also carry important information, such as document domain knowledge, and should be taken into consideration when learning a ranking model. Typically, a ranking model is learned on an entire list that naively combines the features extracted from various fields of a document. In such a way, a trained model cannot adequately and correctly incorporate the domain knowledge and the field contributions. Moreover, an empirical research about how field-based features affect the ranking performance is lacking. Therefore, this study analyses whether using field-based features are more effective than using the naively combined features for learning ranking models. The following questions are the main focus of our research:

- How do the field features behave in comparison to the naively combined features when used to learn ranking models?
- How do the fields differ in their contributions to the development of a ranking model?
- How effective is the aggregation of models built from different fields?

The Field Learning to Rank (fLTR) approach is presented to address these research concerns. On two sizable benchmark datasets that are available to the public, experiments are performed utilizing eight classic LTR algorithms. The features are categorized according to the fields from which they are extracted, and using the field-based features, a number of ranking models are learned. To evaluate the ranking models, three widely adopted assessment

* Corresponding author at: School of Computer Science, Zhongyuan University of Technology, Zhengzhou 450007, China.
E-mail addresses: huayangchn@gmail.com (H. Yang), tcg@uevora.pt (T. Gonçalves).

https://doi.org/10.1016/j.asoc.2023.110183
1568-4946/

metrics are used, namely Precision, MAP (mean average precision), and NDCG (normalized discounted cumulative gain). The findings indicate that using field-based features to train models improves the ranking results. The significant contributions of this paper are, in brief, as follows:

- For all five types of field features studied, their contribution to the ranking performance is compared in depth using three widely adopted evaluation measures. In all three measures, the models using field features show competitive results when compared to the baselines that use naively combined features.
- Two aggregation methods, along with different normalization techniques, are employed to investigate the effectiveness of aggregating results from various fields. In contrast to the baselines which use naively combined features, aggregating results from various fields models presents superior performance.
- Pearson's correlation is used to analyze the correlations among the field features. The deep discussion demonstrates the possibility of joining highly correlated features when naively combining features from various fields. Experiments are carried out to further understand why field-based ranking models can outperform the ones using naively combined features.

This work builds upon the preliminary work presented in [6]. This manuscript summarizes the classic and the newly appeared algorithms according to each LTR approach and reviews and discusses deeply the impact of the text fields in the IR. The preliminary work only evaluates the results with NDCG; this one provides further evaluation on two other measures: Precision and MAP. Moreover, the LTR algorithms are thoroughly compared cross the datasets by different fields and the observations are compared with researchers previous findings.

The rest of the paper is organized as follows: Section 2 reviews the related work and Section 3 describes the proposed approach. Then, Section 4 presents the conducted experiments and analyzes the results and Section 5 further discusses the observations found. Finally, Section 6 draws the conclusions and pinpoints possible future work.

2. Literature review

In this section, the classic and recently appeared LTR approaches are reviewed; the field's impact in IR is surveyed and summarized.

2.1. Learning to Rank approaches

Learning to Rank, an approach used in the Information Retrieval research field, uses Machine Learning techniques to construct ranking models [7]. Queries, related documents and relevance assessments are typically the fundamental components when applying LTR approaches. A LTR model is built on the training dataset; by applying this trained model in a retrieval system, a sorted list of documents is generated according to their relevance to the given testing query (information need) [7].

Different LTR algorithms use various hypotheses, define various input/output spaces, and apply various loss functions. Generally, these algorithms can be classified into three types: (1) Pointwise: the pointwise approach defines the LTR problem as a regression or classification problem, and interprets the relevance degrees as numerical or ordinal scores; (2) Pairwise: the ranking models are trained by employing documents pairs as the training instances; (3) List-wise: the ranking models are trained by employing the entire documents list related to a query as instances [7]. Table 1 summarizes the classic and newly appeared LTR algorithms according to each type.

2.2. Field impact in IR

In recent years, researchers have explored the use of knowledge existing in the texts' field for IR. According to the methods employed to explore the fields, the models proposed can be classified as: IR models [3,28–37], LTR models [4,38–40], neural networks (NN) models [1,5,37,41–44], and others like the statistical analysis [45–47]. Table 2 provides an overview of these research works. Their explored fields and the employed methods are summarized.

An extended version of the standard language model was suggested by Ogilvie and Callan [28]. They used a linear combination of the probabilities of search terms derived from various document fields, such as the in-link content, title, large font content (headings), image alternate content and full text, to score a structured document. The researchers found that the in-link text, title and full text scored better as representation of the document compared to the other document fields. A field relevance model aiming to study the field weighting strategy for structured document retrieval was proposed by Kim and Croft [31]. Three datasets with the corresponding field information were used in their experiments: the IMDB dataset, which contained information about the title, year, and release of the data; the TREC2005 Enterprise dataset, which contained information about the subject, body, and receiver; the Monster dataset, which contained information about the resume's heading, summary, and position. The results showed that the baselines were improved by using the field weighting techniques. Jimmy et al. [3] studied four fields of the structured documents and found that retrieval performance was occasionally improved when the titles were boosted. A fielded sequential dependency model was presented by Zhiltsov et al. [32]. Five entity fields were tested, and field weighting was effective for a variety of queries in ad hoc entity retrieval. Yulianti and Rahadianti [33] incorporated field information into IR for the subject headings identification. Titles, abstracts, and titles combined with abstracts were studied. By utilizing the word position (field features), Hammache and Boughanem [34] extended the language models for IR. Experiments on the search efficiency were conducted, and the title field was studied; the results showed that long queries outperformed short ones across all retrieval models on three testing datasets using precision as performance measure. Ueda et al. [35] employed structured data and considered five fields of a scientific paper. The field effect on retrieval performance was investigated by deleting the field features from the aggregated features gradually. Within the explored fields, in all three considered evaluation measures, the "abstract" field contributed the most, and the other four fields ("conclusions", "methods", "results", and "background") presented different contributions. They also found that "background" and "methods" are positively correlated.

Fernando Diaz [39] used features labeled by experts to build LTR models. Various fields were explored, such as anchor, tile, body or PageRank. The outcomes demonstrated that employing feature labels worked better than the baselines. The efficiency of the LTR method for ranking entities was examined by Chen et al. [4]. They used multi-field representations to describe entities, showing different performances of field features for various kinds of searches. In the study of the LTR for multi-label text classification, Azarbonyad et al. [40] examined the efficiency of the body, title, and other fields and found that the title field was more informative compared to the others.

Jointly using an improved convolutional neural network and a latent semantic model, Shen et al. [1,42] investigated the body and title fields. The results showed that the title field performed better than the body field. Employing a combined ranking model, Mitra et al. [43] examined body text extracted from unprocessed

Table 1
Categorization and representative examples of LTR algorithms. They are classified as pointwise, pairwise, or listwise approaches. The original algorithm name is used and the proposed year is listed in the brackets.

Pointwise	Pairwise	Listwise
MART (2001) [8]	RankSVM (2000) [12]	ListNet (2007) [19]
Random Forest (2001) [9]	RankBoost (2003) [13]	Coordinate Ascent (2007) [20]
PRank (2002) [10]	RankNet (2005) [14]	AdaRank (2007) [21]
McRank (2007) [11]	LambdaMART (2010) [15]	DLCM (2018) [22]
	DirectRanker (2019) [16]	DeepQRank (2020) [23]
	PairRank (2021) [17]	SetRank (2020) [24]
	DeepPLTR (2021) [18]	PiRank (2021) [25]
		PoolRank (2021) [26]
		ListMAP (2022) [27]

Table 2
Overview of the fields investigated within the scope of information retrieval. IR, LTR, and NN stand for information retrieval, learning to rank, and neural network, respectively.

Ref.	Method	Explored fields
[28]	IR model	full text, in-link text, title, image alternate text, large font text
[29,30]	IR model	body, title, extracted title, combined title (extracted title field and the title field)
[31]	IR model	fields selected based on the datasets
[32]	IR model	name, attributes, categories, similar entity names, related entity names
[3]	IR model	title, meta, headers, body
[33]	IR model	title, abstract, the combination (title and abstract)
[35]	IR model	abstract, background, method, result, conclusion
[36]	IR model	title, body, url, etc.
[37]	IR model	title, description, hybrid field
[38]	LTR model	body, title, heading, anchor
[39]	LTR model	title, anchor, body, etc.
[4]	LTR model	RDF (resource description framework) fields
[40]	LTR model	title, body
[41]	NN model	title field of the Web documents
[1,42]	NN model	title, body
[43]	NN model	body text from raw HTML content
[5]	NN model	title, url, body, anchor, clicked queries
[44]	NN model	title, body
[47]	statistical analysis	title, subtitle, keywords, abstract, etc.

HTML material. It was demonstrated that the combined model performed better than the baselines or any single used neural network. Using a neural network to learn the representations, Zamani et al. [5] found that it was preferable to score the entire document as a whole rather than producing individual field scores and aggregating.

Wu [45] employed the method of multiple linear regression to get weights for component retrieval systems. The developed weighting strategy performed better than the linear combination method and other experimented data fusion techniques. De Vries et al. [46] employed a variety of statistical analyses to study the literature searches in different fields, such as title, abstract, keywords, full text, etc. Douze et al. [47] conducted experiments with representative end users to investigate the field's representational relevance. Fields like title, abstract, author keywords, subtitle, and other fields are explored. The title field, followed by the abstract, keywords, and subtitle, presented the most representative relevance, and the other fields showed to be of secondary importance.

3. Methodology

This section illustrates our Field Learning to Rank (fLTR) approach.

The state-of-the-art LTR approaches usually do not consider the contribution of the field from where the features are extracted. In these LTR approaches, a ranking model is trained employing naively combined features; then, given a query, the ranking produced by the trained model is returned.

By contrast, the proposed fLTR approach trains a model in two stages. In the first stage, a group of ranking models is trained, each employing a set of features that are only extracted from one field. The retrieved results are ranked using these ranking models, and thus a group of ranking lists is obtained. In the second stage, the results from the ranking lists are aggregated using the score or rank-based aggregation method. The aggregated results are returned as the final result. Therefore, the proposed fLTR approach provides an effective way to investigate the importance of the field features, and also help to further explore ways to promote the ranking results.

Next, we illustrate the fLTR approach with the two stages mentioned above. The substantial notations to be used are described in Table 3. We hypothesize having a dataset, where the training data includes: (1) a set of queries $q_j (j = 1, \ldots, k)$; (2) the related documents $d^{(j)}$ and their corresponding relevance judgments $y^{(j)}$ in regard to the queries; (3) the document fields F_i the features are extracted from. For the testing data, q represents a testing query, and we aim to use the proposed fLTR approach to return a ranking list of documents given the query q.

3.1. Development of the field LTR models

Within the first step, the features are extracted from various fields and classified into n groups. When training a field based ranking model H_i, only the features that are grouped in the F_i field are used. Consequently, a number of n ranking models $(H_1, H_2, \ldots, H_n)$ are developed. Using the field-based model H_i, a ranking result h_i is produced given a testing query q. The query-document score is expressed as $s_{F_i}(d)$, meaning the score obtained for document d $(d \in D)$ related to a testing query q using the model H_i.

3.2. Aggregation in the fLTR approach

Within the second step, using the aggregation function $G[x]$, the ranking results $(h_1, h_2, \ldots, h_n)$ obtained by each fLTR model

Table 3
Notations for the proposed fLTR method.

Notation	Meaning
D	data collection
d	a document in D
q_j	query contained in the training dataset, j=1...k
$d^{(j)}$	associated document to query q_j in the training dataset
$y^{(j)}$	corresponding relevance judgment to document and query pair ($m^{(j)}$, q_j)
q	testing query
m	number of fields of document d
n	number of fields used for learning models, $n < m$
F_i	the ith field of document d
H_i	model learned using field F_i *information*
h_i	result obtained with model H_i for query q
$s_{F_i}(d)$	score of document d obtained using model H_i *regard to query* q
$G[x]$	aggregation algorithm
S	aggregation score
H	final result

Table 4
Statistics of the experimental datasets.

Dataset	Original features	Queries	Labeled query-document pairs
MQ2007	46	1,700	69,623
MQ2008	46	800	15,211

are aggregated. The aggregation score $S(d)$ is expressed as:

$$S(d) = G[\ s_{F_1}(d), \ldots, s_{F_i}(d), \ldots, s_{F_n}(d)\ ]$$

Assuming each field is assigned a weight of w_i, the weighted result for document d is expressed as:

$$S(d) = G[\ w_1 s_{F_1}(d), \ldots, w_i s_{F_1}(d), \ldots, w_n s_{F_n}(d)\ ]$$

Simply, assuming $G[x]$ is a linear function, $S(d)$ can be defined as:

$$S(d) = \sum_{i=1}^{n} w_i s_{F_1}(d) \quad (1)$$

The final result H is produced by applying this strategy to the scores obtained from each ranking list ($h_1, h_2, \ldots, h_n$).

4. Experiments and results

This section, first describes the datasets used for the experiments; next, it presents the baselines and the fLTR and the aggregation rankers built; then, it explains the adopted evaluation measures and, finally, it details the evaluation of the rankers and the results obtained.

4.1. Datasets

We apply our fLTR approach and conduct the experiments using Microsoft LETOR 4.0 [48] benchmark datasets, a data collection for LTR provided by Microsoft Research. The two datasets that compose it, MQ2007 and MQ2008, use the query set from Million Query track of TREC2007 and TREC2008 [48,49]. Both datasets use the Gov2 web page collection which includes about 25 million pages [48].

Each dataset includes: (1) a number of queries and the corresponding retrieved documents obtained through real search activities; (2) a set of standard features; (3) relevance judgments annotated by annotators. The relevance between a query and a document is labeled with a digital number, with 2 representing highly relevant, 1 for relevant, and 0 for not relevant. The statistics of the two datasets are presented in Table 4.

Each dataset is already split into 5 folds, so a 5-fold cross validation strategy is adopted in our experiments. In each fold, there are three subsets for learning: training set, validation set and testing set. The detailed description of these subsets is presented in Table 5.

Table 5
Training/Validation/Testing 5 fold split of MQ2007 and MQ2008.

	MQ2007				
	fold1	fold2	fold3	fold4	fold5
training	42,158	41,958	41,320	41,478	41,955
validation	13,813	13,652	14,013	14,290	13,855
testing	13,652	14,013	14,290	13,855	13,813
	MQ2008				
	fold1	fold2	fold3	fold4	fold5
training	9,630	9,404	8,643	8,514	9,442
validation	2,707	2,874	2,933	3,635	3,062
testing	2,874	2,933	3,635	3,062	2,707

Table 6
The selected fields and features in the experiments.

Feature	Field				
	body	anchor	title	url	wholedoc
TF	✓	✓	✓	✓	✓
IDF	✓	✓	✓	✓	✓
TF*IDF	✓	✓	✓	✓	✓
DL	✓	✓	✓	✓	✓
BM25	✓	✓	✓	✓	✓
LMIR.ABS	✓	✓	✓	✓	✓
LMIR.DIR	✓	✓	✓	✓	✓
LMIR.JM	✓	✓	✓	✓	✓

To adapt the datasets to our experiments, a selection is made from the original 46 types of standard features [48]. The selection can be described as follows: (1) all the features are classified into six groups based on which field they are extracted from, namely *url*, *anchor*, *title*, *body*, *wholedoc* (whole document), or *other place*; (2) the features extracted from *other place* provide no fields information, and they are eliminated in our experiments. So, features *number of child page*, *number of inlink*, *number of outlink*, and *pagerank* are removed; (3) to make sure that the comparison among various fields is fair, the same number and uniformity of the features are kept for each field. So, features *length of url* and *number of slash in url* are removed. As a result, five fields are taken into account, and each field contains eight types of features. Table 6 details the features per field included in the experiments (40 in total).

4.2. Building ranking models

We consider all three categories of LTR algorithms (point-wise, pair-wise, and list-wise) in our experiments and experiment on

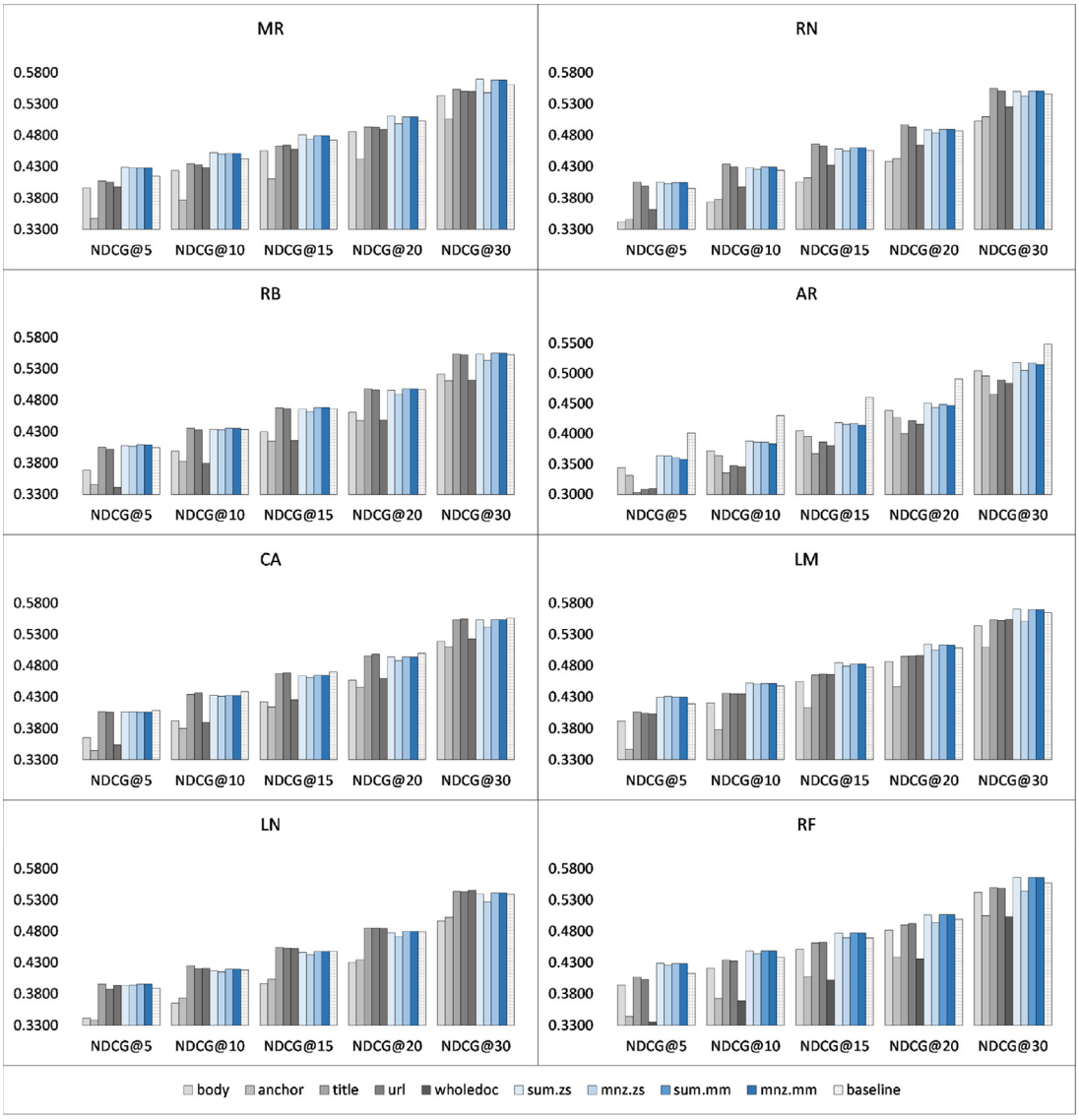

Fig. 1. NDCG evaluations on MQ2007 dataset: comparisons between the baselines, the models using single field features, and the aggregated models.

8 state-of-the-art ones: MART (MR) [8], Random Forests (RF) [9], RankNet (RN) [14], RankBoost (RB) [13], LambdaMART (LM) [15], AdaRank (AR) [21], Coordinate Ascent (CA) [20], and ListNeT (LN) [19].

Baseline models, using one of the 6 chosen LTR algorithms, are trained using the feature list that naively joins all the 40 features.

When using the proposed fLTR approach, we first train a group of field-based LTR models and then aggregate the results (see Section 3). During the first phase, a different fLTR model is trained using the features from a specific field (see Section 3.1), totaling 5 trained models for each algorithm. By testing 8 LTR algorithms, a total of 40 fLTR models are built. On the second phase (Section 3.2), CombMNZ and CombSUM are chosen as possible aggregation algorithms [50]; we set w_i to 1 in Eq. (1) to keep the same weights for all five fields.

Also, before aggregating the results, Z-score and Min–Max are used as possible normalization algorithms [51]. By using the aggregation and normalization algorithms crossly, a set of 32 fLTR aggregation models (8 LTR, 2 aggregation and 2 normalization algorithms) are built for each experimental dataset. Consequently, on each experimental dataset (MQ2007 and MQ2008), 8 baselines and 72 fLTR models are developed.

4.3. Evaluation measures

To evaluate the performance of our proposed method, we calculate three widely adopted evaluation measures in LTR, namely Precision [52], MAP [53], and NDCG [54].

4.3.1. Precision at Rank n (P@n)

Precision is one fundamental metric and has been expanded and tailored to various kinds of tasks [52]. Given a query, Precision at Rank n (P@n) is the proportion of the relevant documents that are retrieved among the top-n ranking list [55]. The measure is defined as

$$\text{P@n} = \frac{r}{n}$$

meaning that, at position n of the ranked list, r relevant documents are retrieved. The averaged P@n value over all the queries is report in our experiments.

In the MQ2007 and MQ2008 datasets, the judgments for the query and document pair are provided in three ratings: *irrelevant*, *slightly relevant* and *relevant*, but only a binary judgment (*irrelevant* or *relevant*) is required for P@n. So, in our evaluation using P@n metric, *irrelevant* judgments in the original dataset are

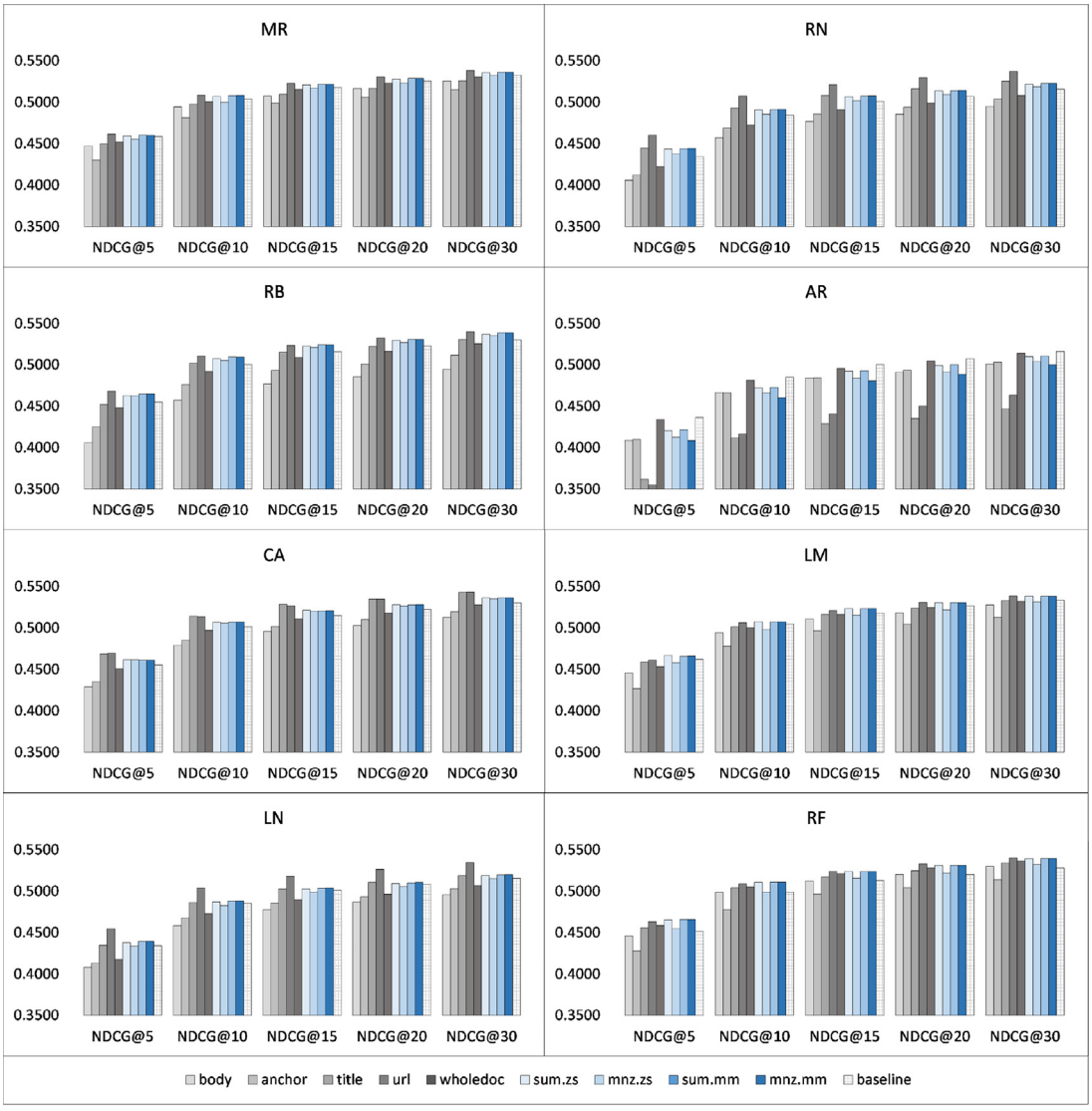

Fig. 2. NDCG evaluations on MQ2008 dataset: comparisons between the baselines, the models using single field features, and the aggregated models.

treated as *irrelevant* and *slightly relevant* and *relevant* are treated as relevant.

4.3.2. Mean average precision (MAP)

Mean Average Precision (*MAP*) is another commonly used measure and has shown to have good discrimination and stability [56]. Given a single query, the Average Precision (*AP*) is the average score obtained after each relevant document is retrieved for a set of top documents [57]. Conceptually, the *MAP* value in an IR system is the arithmetic mean of the *AP* values for a set of query topics. It is defined as [58]

$$\text{MAP} = \sum_{q=1}^{Q} \frac{\text{Average Precision}}{Q}$$

where a single query q belongs to Q, the total number of queries.

4.3.3. Normalized discounted cumulative gain (NDCG)

Besides the commonly used evaluation measures mentioned above, we have also observed the performance of the rankers in terms of NDCG [54]. NDCG is one of the best evaluation measures when using ML approaches for ranking [56]. *NDCG@n* refers to the NDCG value obtained at position n, which can be conceptually described as

$$\text{NDCG@n} = \frac{\text{DCG}_n}{\text{IDCG}_n}$$

where *DCG* (Discounted Cumulative Gain) is the gain accumulated over the results from the top to the bottom in a ranking list. This measure penalizes highly relevant documents appearing lower in the ranking list. IDCG is the ideal discounted cumulative gain, where documents are ideally ranked according to their relevance and high relevant document has an early appearance.

All ranking models were built with the RankLib[1] tool. Using the LETOR benchmark dataset configuration [48], experiments were conducted using five-fold cross-validation. Models were also fine tuned using the validation dataset. The results of the average scores on the testing dataset are reported at the positions of 5, 10, 15, 20, and 30 for P@n, MAP@n and NDCG@n.

4.4. Results

In this section, the results observed on NDCG@n are discussed in detail. Similar findings can be observed on P@n and MAP@n.

[1] https://sourceforge.net/p/lemur/wiki/RankLib/

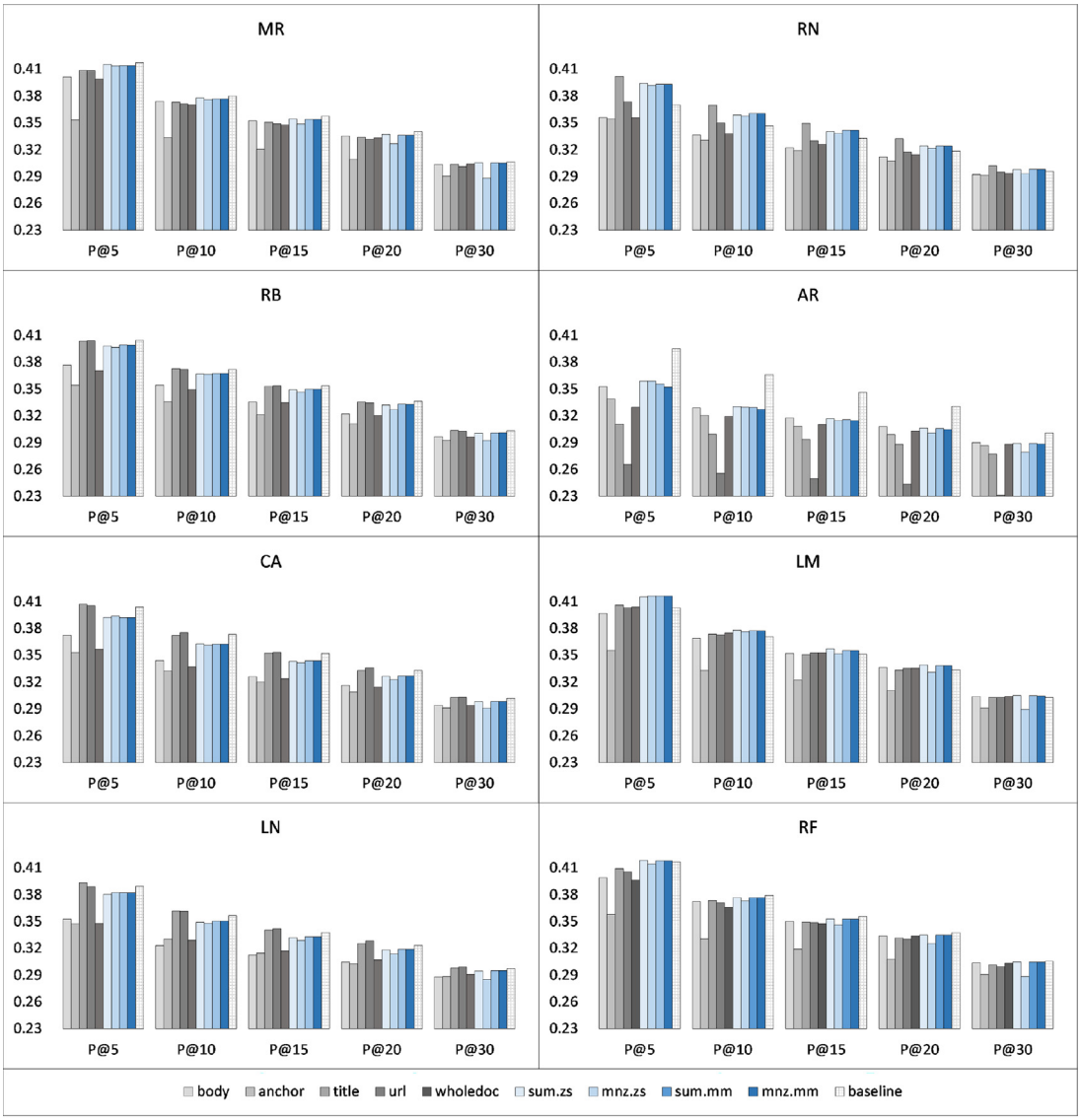

Fig. 3. Precision evaluation on MQ2007 dataset: comparisons between the baselines, the models using single field features, and the aggregated models.

Figs. 1 and 2 present NDCG results on MQ2007 and MQ2008 datasets, respectively.

The observations on the MQ2007 dataset are: (1) the fLTR models developed with *title* information outperform all baselines on three LTR algorithms (RN, RB, and LN) and obtain competitive results on algorithm CA; (2) using *url* information exceed the baselines on algorithms RN, LN (NDCG@10, 15, 20, and 30) and RB (NDCG@15), yielding competitive result on CA; (3) when using field *whole document*, the developed fLTR models exceed the baselines on algorithm LN; (4) *body* and *anchor* information fail to exceed the baselines in all instances; (5) the aggregation models using the CombSUM approach outperform the majority of baselines in two algorithms (RB and LN) and all baselines in four algorithms (MR, LN, RN, and RF). In most circumstances, the CombSUM approach performs better with MinMax than with Z-score normalization; (6) the CombMNZ aggregation with Min–Max normalization, outperforms the baselines on nearly all algorithms. When Z-score normalization is used, the models outperform the baselines on algorithms MR, LM, and RF in most circumstances, and on RN, RB, and LN in certain cases.

The observations on the MQ2008 dataset using the NDCG evaluation metrics are: (1) the fLTR models using *url* information exceed all baselines excluding algorithm AR and LN on P@5; (2) the fLTR models generated with *title* information outperform all baselines in the majority LTR algorithms (RN, RB, CA, LN, and RF); (3) when using field *whole document*, the developed fLTR models surpass the baselines on LN and RF; (4) the models utilizing *body* field exceed the baselines on RF (NDCG@20 and NDCG@30); (5) the fLTR models built using *anchor* information fail to exceed the baselines in all circumstances; (6) the aggregation models using the CombSUM approach outperform all baselines on all algorithms except AR. The CombSUM approach presents similar performance on MinMax and Z-score normalization; (7) the CombMNZ aggregation with Min–Max normalization outperforms the baselines on all algorithms except AR. When Z-score normalization is used, the models outperform the baselines on algorithms RN, RB, CA, and RF in most circumstances, and on MR, LM, and LN in certain cases.

In addition to these direct findings, the winning number (*WN*) measure [53] was used to count the winnings across both datasets and provide insight into the overall ranking performance on NDCG. Eq. (2) presents the measure where M denotes the measure, i and k denotes the index of an algorithm, j denotes the index of the datasets, and $M_i(j)$ denotes the performance of i-th algorithm on j-th dataset.

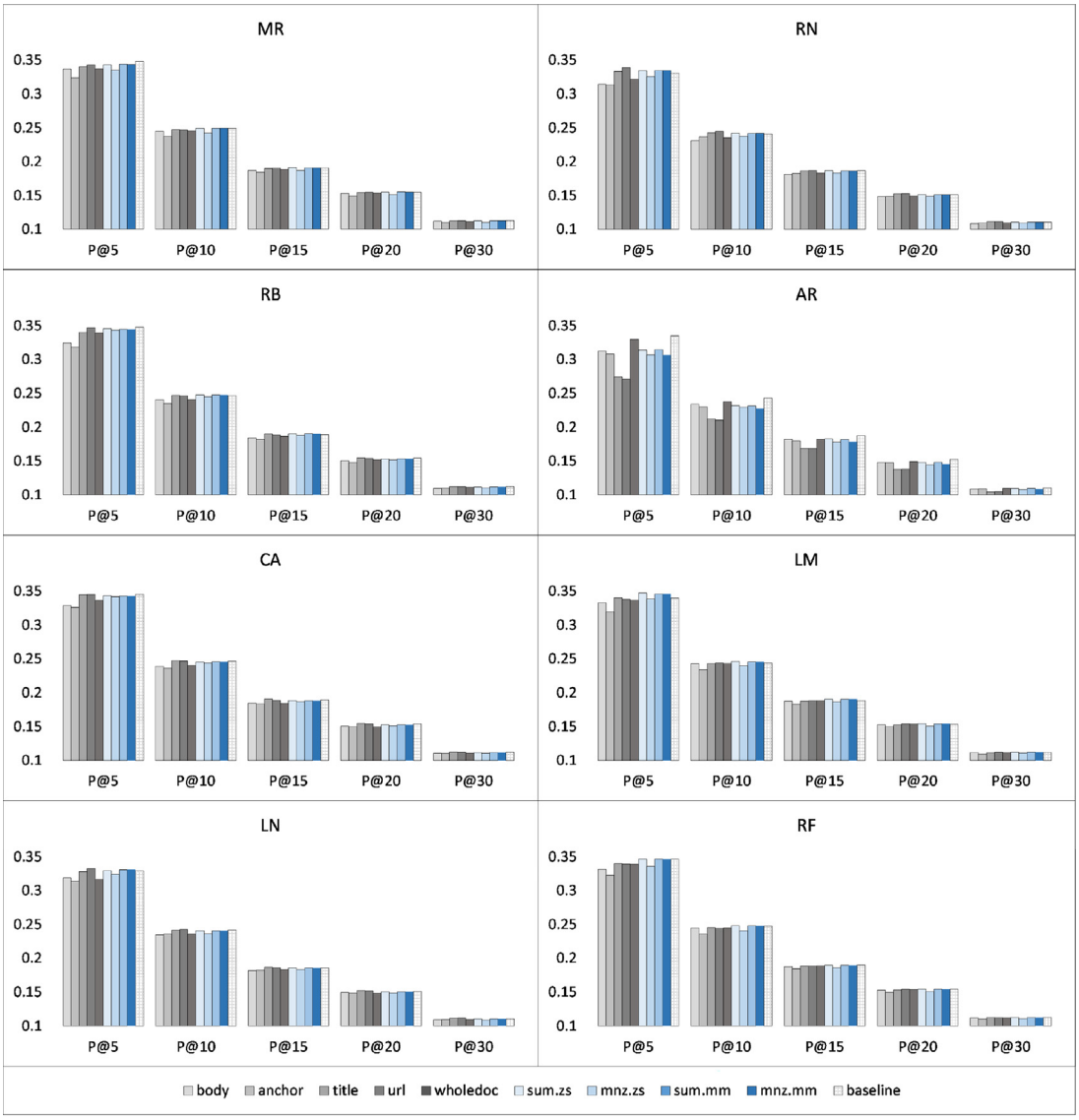

Fig. 4. Precision evaluation on MQ2008 dataset: comparisons between the baselines, the models using single field features, and the aggregated models.

$$WN_i = \sum_{j=1}^{n}\sum_{k=1}^{m} I_{\{M_i(j)>M_k(j)\}}$$

$$I_{\{M_i(j)>M_k(j)\}} = \begin{cases} 1 & \text{if } M_i(j) > M_k(j) \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

The winning number measure is used on the five investigated fields, the four aggregations, and the eight standard learning to rank algorithms over the MQ2007 and MQ2008 datasets. As the results shown in Table 7, the *WN* is calculated in two dimensions: how the fields and their aggregations perform across different LTR algorithms, and vice-versa: rows indicate the winning number of a field or an aggregation across all LTR algorithm and columns indicate the winning number of different LTR algorithm across all fields and their aggregations. For the rest of the article, aggregations and normalizations are abbreviated for ease of notation: CombSUM is noted as sum, CombMNZ as mnz, Z-score as zs, and MinMax as mm. For example, sum.zs indicates using the CombSUM as the aggregation and Z-score as the normalization.

The following observations can be obtained from Table 7:

Table 7
The winning number (WN) with NDCG evaluation: fields, aggregations, and LTR algorithms on MQ2007 and MQ2008 datasets.

	MR	RN	RB	AR	CA	LM	LN	RF	WN
body	0	0	0	0	0	0	0	4	4
anchor	0	0	0	0	0	0	0	0	0
title	0	10	7	0	10	0	10	10	47
url	10	10	10	0	10	8	10	10	68
wholedoc	0	0	0	0	0	0	10	10	20
sum.zs	10	10	10	0	10	10	10	10	70
mnz.zs	6	10	10	0	10	6	2	10	54
sum.mm	10	10	10	0	10	10	10	10	70
mnz.mm	10	10	10	0	10	10	10	10	70
WN	46	60	57	0	60	44	62	74	–

- Overall field performance. Among the five investigated fields, *url* and *title* are significantly better than the other three: *url* scores the best with a winning number of 68 and *title* presents a winning number of 47. In contrast, the other three fields score much lower: the *wholedoc* obtains a winning number of 20 and the *body* field obtains 4; *anchor* field scores the worst in all five fields and loses in all cases. Our

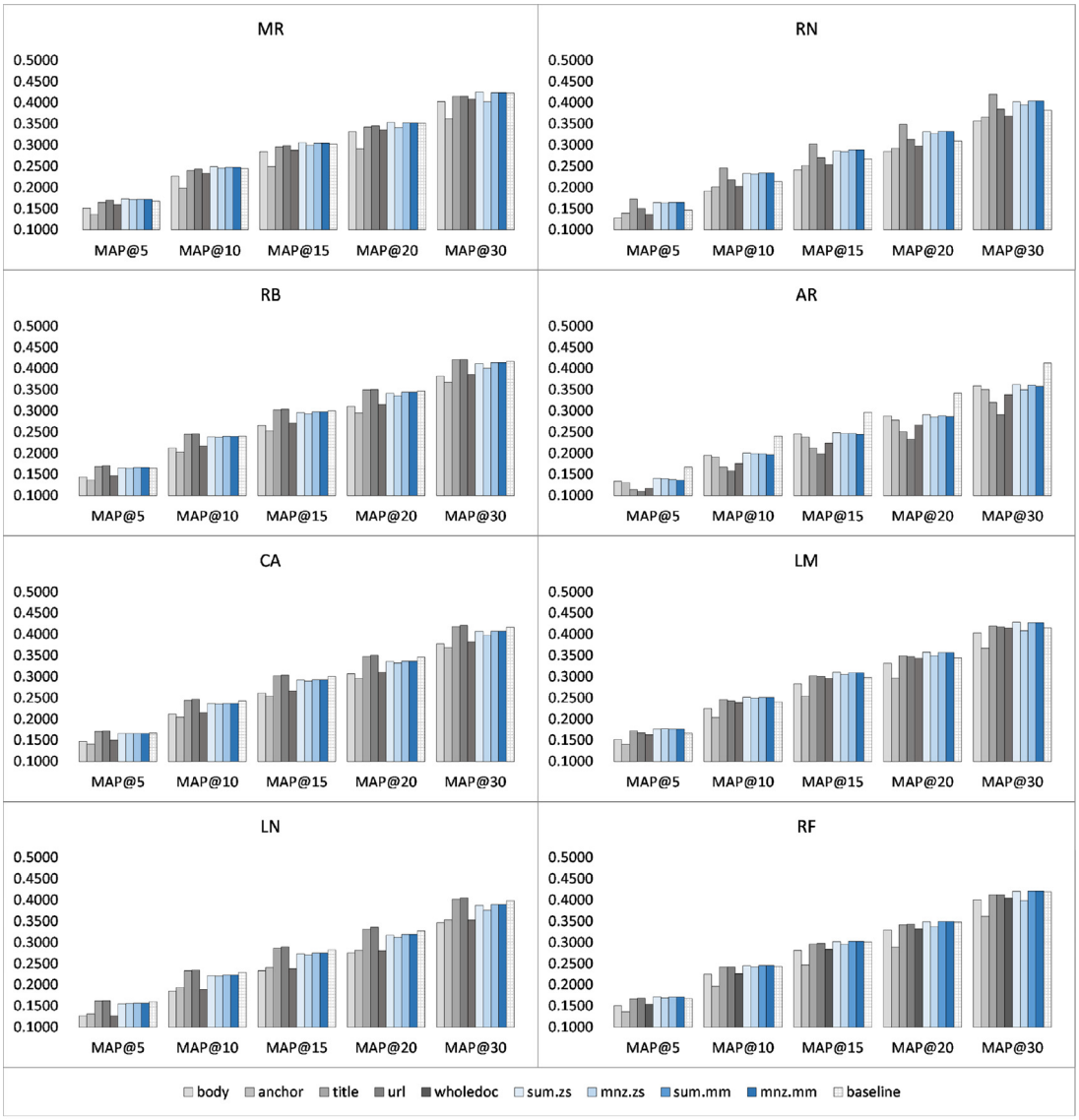

Fig. 5. MAP evaluations on MQ2007 dataset: comparisons between the baselines, the models using single field features, and the aggregated models.

findings are consistent with the literature indicating that the *title* field is more effective than *body* in improving the ranking results [1,31,38,40,59].

- Overall aggregation performance. For the four aggregation methods, *sum.zs*, *sum.mm* and *mnz.mm* score the best, with a winning number of 70 each. Compared to these three aggregations, *mnz.zs* presents a slightly worse performance obtaining a winning number of 54. For additional insight into the results and interestingly, CombSUM seems to be more robust since it presents similar and stable results (70 versus 70) with the two different normalization techniques (Z-score and Min–Max), while ComMNZ scores much higher with Min–Max than with Z-score (70 versus 54). These findings are also consistent with a recent work by Ueda et al. [35] in biomedical retrieval, where the ranker built with field-level aggregation presented better retrieval performance than the one using a single field.
- Overall LTR algorithm performance. It is also interesting to investigate how the three kinds of LTR algorithms perform in the proposed fLTR approach. With regard to the pointwise LTR algorithms, RF (Random Forest) outperforms MR (MART) algorithm, with a competing winning number of 74 to 46; for the pairwise LTR algorithms, RN (RankNet) and RB (RankBoost) obtain a winning number of 60 and 57, respectively; they perform more effectively than LM (LambdaMART) which obtains 44; considering the listwise algorithms, the finest performance comes from LN (ListNet) with a winning number of 62, followed by CA (Coordinate Ascent) with a number of 60, with AR (AdaRank) presenting the worst performance and failing to perform better than the baseline in all experiments.

Following the same evaluation process conducted using the NDCG metric, the results evaluated with P@n are presented in Figs. 3 and 4 and MAP@n are presented in Figs. 5 and 6. From these figures, we can observe similar results to the ones found for NDCG@n.

Similarly, we compare the winning numbers in terms of P@n and MAP@n, and the results are shown in Tables 8 and 9, respectively. The fields and the aggregations present similar results to the NDCG@n metric. Concerning the LTR algorithms, LambdaMART (LM) shows better performance than RankBoost (RB), with a winning number of 53 to 14 on P@n, and 50 to 18 on MAP@n; Coordinate Ascent (CA) performs better than ListNet (LN)

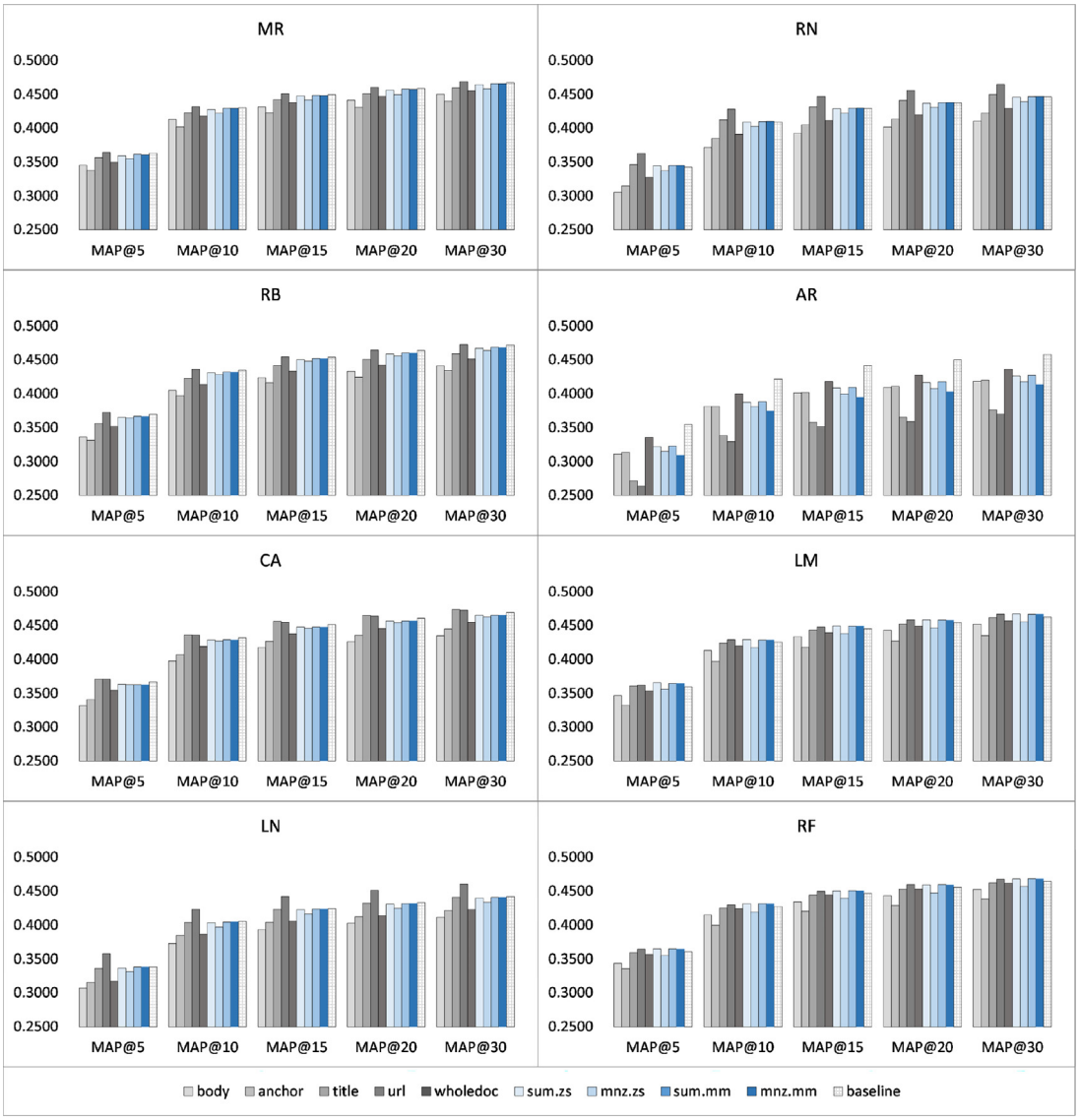

Fig. 6. MAP evaluations on MQ2008 dataset: comparisons between the baselines, the models using single field features, and the aggregated models.

Table 8
The winning number (WN) with Precision evaluation: fields, aggregations, and LTR algorithms on MQ2007 and MQ2008 datasets.

	MR	RN	RB	AR	CA	LM	LN	RF	WN
body	0	0	0	0	0	3	0	0	3
anchor	0	0	0	0	0	0	0	0	0
title	0	9	6	0	8	3	8	0	34
url	0	7	2	0	9	8	9	1	36
wholedoc	0	0	0	0	0	6	0	0	6
sum.zs	3	8	2	0	0	10	3	4	30
mnz.zs	0	4	0	0	0	3	0	0	7
sum.mm	3	7	2	0	0	10	3	4	29
mnz.mm	3	8	2	0	0	10	3	4	30
WN	9	43	14	0	17	53	26	13	–

Table 9
The winning number (WN) with MAP evaluation: fields, aggregations, and LTR algorithms on MQ2007 and MQ2008 datasets.

	MR	RN	RB	AR	CA	LM	LN	RF	WN
body	0	0	0	0	0	0	0	0	0
anchor	0	0	0	0	0	0	0	0	0
title	0	10	5	0	10	6	5	0	36
url	6	10	10	0	10	10	10	6	62
wholedoc	0	0	0	0	0	0	0	0	0
sum.zs	5	7	1	0	0	10	0	10	33
mnz.zs	2	5	0	0	0	4	0	1	12
sum.mm	5	10	1	0	0	10	0	10	36
mnz.mm	5	10	1	0	0	10	0	10	36
WN	23	52	18	0	20	50	15	37	–

with a winning number of 20 to 15 on MAP@n. The algorithms present similar performances to NDCG@n in all the other cases.

To sum up, the experimental results show that the proposed fLTR approach is very promising and has advantages over the baselines in all three evaluation measures (P@n, MAP@n, and NDCG@n).

5. Discussion

The results presented in Section 4.4 show that the ranking models developed using the proposed fLTR approach are able to exceed the baselines in the majority of cases. Meanwhile, as described in Section 3, the fLTR approach uses fewer features

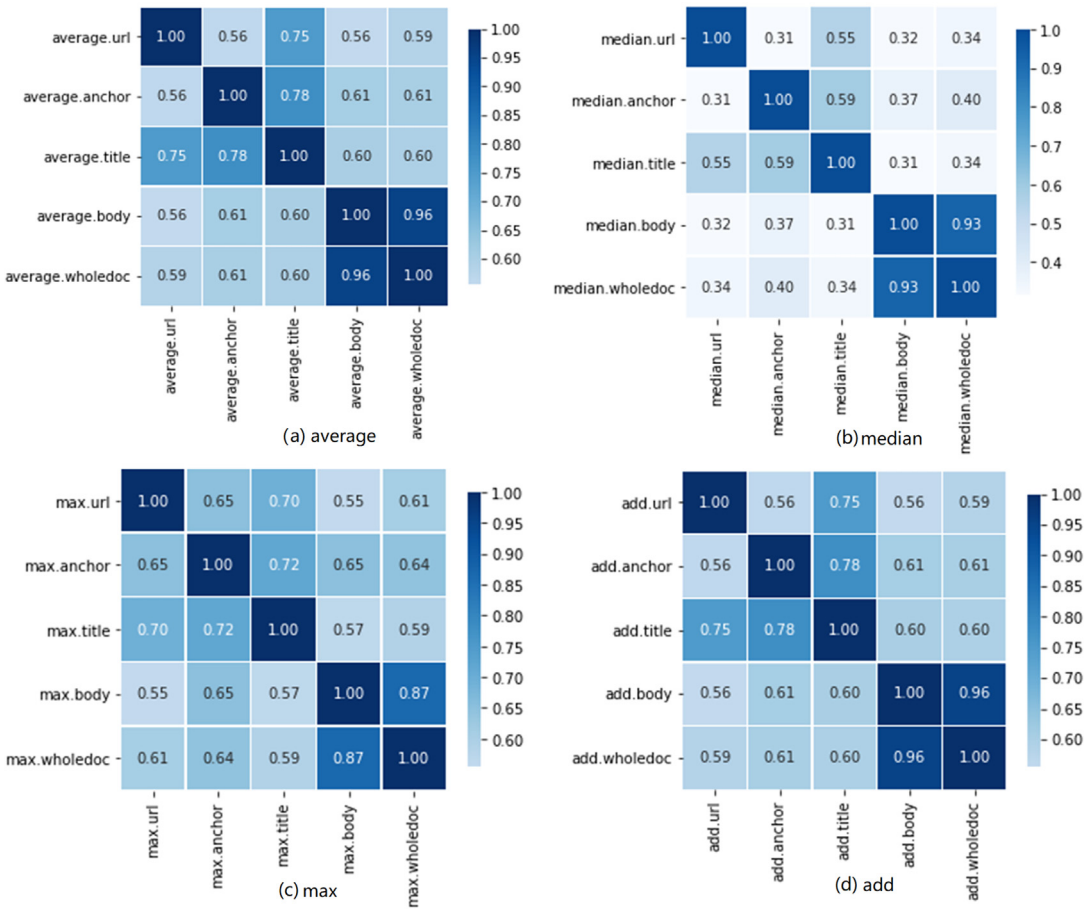

Fig. 7. Comparing Pearson's correlations. The *average*, *median*, *maximum*, and *sum* values are computed based on the MQ2008 data collection's field features.

compared to the state-of-the-art LTR ones. It is interesting to investigate why using fewer features can produce better results.

When applying ML techniques, naively joining strongly correlated features can result in decreased performance; more diversified and discriminating features with little correlation are desired [60]. Following this idea, statistical tests were carried out on the MQ2008 data collection. Pearson's correlations were calculated to study feature interconnections. Fig. 7 presents the heatmaps for the *average*, *median*, *maximum*, and *sum* feature pairs values. It can be observed that there are high correlations in all 4 measure for the following feature pairs: *title-anchor*, *title-url*, and *body-whole document*.

To learn more about how these correlated features impact ranking results, the following two types of models were trained and compared: (1) using the features from a single field; (2) using the combined features that are highly correlated. Based on the results observed in Fig. 7, two highly correlated feature groups were compared: *title-anchor* (or *url*) and *body-wholedoc*.

The models trained on the MQ2008 data collection were compared using the NDCG metric. Table 10 shows the results. For the *title-anchor* (or *url*) group, in all five NDCG measurements, using *title* individually present better performance than using *title* and *anchor* (or *url*). Also, using the *body* or the *wholedoc* features alone better results are obtained than the joining the features together.

Besides the effectiveness presented by the built rankers in Section 4.4, the analysis in this section offer insights regarding the rationale of the proposed filter approach. We discuss that naively combining all features extracted from various fields can result in high correlations that have a negative effect on the ranking results. Comparatively, the fLTR technique, using field-based and fewer features, prevents the probability of combining high correlated features and their straight interference in the LTR process. These observations also support the earlier findings by Fernando Diaz [39].

6. Conclusion and future work

This work investigates the effects of field features in Learning to Rank for Information Retrieval. For this purpose, the Field Learning to Rank (fLTR) technique is proposed. Experiments are performed on two benchmark datasets using eight well-known LTR algorithms. From our findings, the ranking models using the proposed fLTR technique achieve better results than the ones using the state-of-the-art LTR approaches. Our research also shows that different fields of the document contribute differently to the ranking performance. For example, the *url* and *title* fields have more impact than the *anchor*, *body*, and the *whole document* ones. Moreover, our empirical investigation reveals that the features extracted from various fields, such as *url* and *title*, *anchor* and *title*, *body* and *whole document*, exhibit high correlations. Combining strongly correlated features can lead to a decreased model performance; the proposed fLTR technique has the benefit of avoiding this problem.

The document fields present diverse contributions to the ranking performance, and researchers also argue that boosting a specific field is effective in improving the results; for example, [45] refers that the weighting strategy improves the performance of the component retrieval systems and [3] concludes that boosting *title* improves retrieval effectiveness on some IR tasks. Therefore, a good direction for future work is to explore adding different weights to the selected fields.

Table 10
Comparing the ranking performances between the models using highly correlated features and only field features. All models are trained using the LambdaMART algorithm and evaluated with NDCG at different positions.

Compared fields	Features	NDCG				
		@5	@10	@15	@20	@30
title and anchor/url	title	0.4587	0.5014	0.5163	0.5240	0.5331
	title+anchor	0.4471	0.4929	0.5070	0.5143	0.5229
	title+url	0.4465	0.4985	0.5109	0.5184	0.5273
body and wholedoc	body	0.4457	0.4944	0.5108	0.5184	0.5279
	wholedoc	0.4532	0.5002	0.5164	0.5248	0.5318
	body+wholedoc	0.4420	0.4893	0.5064	0.5131	0.5216

CRediT authorship contribution statement

Hua Yang: Conceptualization, Methodology, Software, Visualization, Investigation, Data curation, Writing – original draft. **Teresa Gonçalves:** Conceptualization, Supervision, Data curation, Validation, Writing – reviewing & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The benchmark datasets are publicly available.

References

[1] Y. Shen, X. He, J. Gao, L. Deng, G. Mesnil, A latent semantic model with convolutional-pooling structure for information retrieval, in: Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, 2014, pp. 101–110.
[2] J. Gao, X. He, J.-Y. Nie, Clickthrough-based translation models for web search: from word models to phrase models, in: Proceedings of the 19th ACM International Conference on Information and Knowledge Management, 2010, pp. 1139–1148.
[3] G. Zuccon, B. Koopman, Boosting titles does not generally improve retrieval effectiveness, in: Proceedings of the 21st Australasian Document Computing Symposium, 2016, pp. 25–32.
[4] J. Chen, C. Xiong, J. Callan, An empirical study of learning to rank for entity search, in: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2016, pp. 737–740.
[5] H. Zamani, B. Mitra, X. Song, N. Craswell, S. Tiwary, Neural ranking models with multiple document fields, in: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, 2018, pp. 700–708.
[6] H. Yang, T. Gonçalves, An empirical study of the impact of field features in learning-to-rank method, in: International Conference on Intelligent Data Engineering and Automated Learning, Springer, 2021, pp. 176–187.
[7] T.-Y. Liu, et al., Learning to rank for information retrieval, 3, (3) Foundations and Trends® in Information Retrieval, 2009, pp. 225–331.
[8] J.H. Friedman, Greedy function approximation: a gradient boosting machine, Ann. Statist. (2001) 1189–1232.
[9] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32.
[10] K. Crammer, Y. Singer, Pranking with ranking, in: T.G. Dietterich, S. Becker, Z. Ghahramani (Eds.), Neural Information Processing Systems: Natural and Synthetic, NIPS 2001, December 3–8, 2001, Vancouver, British Columbia, Canada, in: Advances in Neural Information Processing Systems, vol. 14, MIT Press, 2001, pp. 641–647.
[11] P. Li, Q. Wu, C. Burges, Mcrank: Learning to rank using multiple classification and gradient boosting, Adv. Neural Inf. Process. Syst. 20 (2007) 897–904.
[12] T. Joachims, Training linear SVMs in linear time, in: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2006, pp. 217–226.
[13] Y. Freund, R. Iyer, R.E. Schapire, Y. Singer, An efficient boosting algorithm for combining preferences, J. Mach. Learn. Res. 4 (Nov) (2003) 933–969.
[14] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, G. Hullender, Learning to rank using gradient descent, in: Proceedings of the 22nd International Conference on Machine Learning, 2005, pp. 89–96.
[15] Q. Wu, C.J. Burges, K.M. Svore, J. Gao, Adapting boosting for information retrieval measures, Inf. Retr. 13 (3) (2010) 254–270.
[16] M. Köppel, A. Segner, M. Wagener, L. Pensel, A. Karwath, S. Kramer, Pairwise learning to rank by neural networks revisited: Reconstruction, theoretical analysis and practical performance, in: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, 2019, pp. 237–252.
[17] Y. Jia, H. Wang, S. Guo, H. Wang, Pairrank: Online pairwise learning to rank by divide-and-conquer, in: Proceedings of the Web Conference 2021, 2021, pp. 146–157.
[18] K. Yuan, D. Kuang, Deep pairwise learning to rank for search autocomplete, 2021, CoRR abs/2108.04976. arXiv:2108.04976. URL https://arxiv.org/abs/2108.04976.
[19] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, H. Li, Learning to rank: from pairwise approach to listwise approach, in: Proceedings of the 24th International Conference on Machine Learning, ACM, 2007, pp. 129–136.
[20] D. Metzler, W.B. Croft, Linear feature-based models for information retrieval, Inf. Retr. 10 (3) (2007) 257–274.
[21] J. Xu, H. Li, Adarank: a boosting algorithm for information retrieval, in: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2007, pp. 391–398.
[22] Q. Ai, K. Bi, J. Guo, W.B. Croft, Learning a deep listwise context model for ranking refinement, in: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, 2018, pp. 135–144.
[23] A. Sharma, Listwise learning to rank with deep Q-networks, 2020, CoRR, abs/2002.07651 arXiv:2002.07651 URL https://arxiv.org/abs/2002.07651.
[24] L. Pang, J. Xu, Q. Ai, Y. Lan, X. Cheng, J. Wen, Setrank: Learning a permutation-invariant ranking model for information retrieval, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 499–508.
[25] R. Swezey, A. Grover, B. Charron, S. Ermon, Pirank: Scalable learning to rank via differentiable sorting, Adv. Neural Inf. Process. Syst. 34 (2021).
[26] Z. Chen, C. Eickhoff, PoolRank: Max/min pooling-based ranking loss for listwise learning & ranking balance, 2021, CoRR abs/2108.03586. arXiv:2108.03586. URL https://arxiv.org/abs/2108.03586.
[27] S. Keshvari, F. Ensan, H.S. Yazdi, ListMAP: Listwise learning to rank as maximum a posteriori estimation, Inf. Process. Manage. 59 (4) (2022) 102962.
[28] P. Ogilvie, J. Callan, Combining document representations for known-item search, in: Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval, 2003, pp. 143–150.
[29] Y. Hu, G. Xin, R. Song, G. Hu, S. Shi, Y. Cao, H. Li, Title extraction from bodies of HTML documents and its application to web page retrieval, in: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2005, pp. 250–257.
[30] Y. Xue, Y. Hu, G. Xin, R. Song, S. Shi, Y. Cao, C.-Y. Lin, H. Li, Web page title extraction and its application, Inf. Process. Manage. 43 (5) (2007) 1332–1347.
[31] J.Y. Kim, W.B. Croft, A field relevance model for structured document retrieval, in: European Conference on Information Retrieval, Springer, 2012, pp. 97–108.
[32] N. Zhiltsov, A. Kotov, F. Nikolaev, Fielded sequential dependence model for ad-hoc entity retrieval in the web of data, in: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2015, pp. 253–262.
[33] E. Yulianti, L. Rahadianti, Determining subject headings of documents using information retrieval models, Indonesian J. Electr. Eng. Comput. Sci. 23 (2) (2021) 1049–1058.
[34] A. Hammache, M. Boughanem, Term position-based language model for information retrieval, J. Assoc. Inf. Sci. Technol. 72 (5) (2021) 627–642.
[35] A. Ueda, R.L. Santos, C. Macdonald, I. Ounis, Structured fine-tuning of contextual embeddings for effective biomedical retrieval, in: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2021, pp. 2031–2035.

[36] J. Devins, J. Tibshirani, J. Lin, Aligning the research and practice of building search applications: Elasticsearch and pyserini, in: Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, 2022, pp. 1573–1576.
[37] L. Boualili, J.G. Moreno, M. Boughanem, Highlighting exact matching via marking strategies for ad hoc document ranking with pretrained contextualized language models, Inf. Retr. J. 25 (4) (2022) 414–460.
[38] N. Dai, M. Shokouhi, B.D. Davison, Learning to rank for freshness and relevance, in: Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2011, pp. 95–104.
[39] F. Diaz, Learning to rank with labeled features, in: Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval, 2016, pp. 41–44.
[40] H. Azarbonyad, M. Dehghani, M. Marx, J. Kamps, Learning to rank for multi-label text classification: Combining different sources of information, Nat. Lang. Eng. 27 (1) (2021) 89–111.
[41] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, L. Heck, Learning deep structured semantic models for web search using clickthrough data, in: Proceedings of the 22nd ACM International Conference on Information & Knowledge Management, 2013, pp. 2333–2338.
[42] Y. Shen, X. He, J. Gao, L. Deng, G. Mesnil, Learning semantic representations using convolutional neural networks for web search, in: Proceedings of the 23rd International Conference on World Wide Web, 2014, pp. 373–374.
[43] B. Mitra, F. Diaz, N. Craswell, Learning to match using local and distributed representations of text for web search, in: Proceedings of the 26th International Conference on World Wide Web, 2017, pp. 1291–1299.
[44] Y. Yang, Y. Qiao, J. Shao, X. Yan, T. Yang, Lightweight composite re-ranking for efficient keyword search with BERT, in: Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, 2022, pp. 1234–1244.
[45] S. Wu, Linear combination of component results in information retrieval, Data Knowl. Eng. 71 (1) (2012) 114–126.
[46] B.B.P. de Vries, M. van Smeden, F.R. Rosendaal, R.H. Groenwold, A comparison between full text mining and searching in title, abstract and keywords for systematic reviews of epidemiological practice, Bas Penning de Vries 121 (2020) 15.
[47] L. Douze, S. Pelayo, N. Messaadi, J. Grosjean, G. Kerdelhué, R. Marcilly, et al., Designing formulae for ranking search results: Mixed methods evaluation study, JMIR Hum. Fact. 9 (1) (2022) e30258.
[48] T. Qin, T.-Y. Liu, Introducing LETOR 4.0 datasets, 2013, arXiv preprint arXiv:1306.2597.
[49] J. Allan, B. Carterette, J.A. Aslam, V. Pavlu, B. Dachev, E. Kanoulas, Million query track 2007 overview, Technical Report, Massachusetts Univ Amherst Dept of Computer Science, 2007.
[50] E.A. Fox, J.A. Shaw, Combination of multiple searches, NIST Special Publ. SP 243 (1994).
[51] D. Singh, B. Singh, Investigating the impact of data normalization on classification performance, Appl. Soft Comput. 97 (2020) 105524.
[52] B. Carterette, Precision and recall, in: Encyclopedia of Database Systems, Springer US, Boston, MA, 2009, pp. 2126–2127, http://dx.doi.org/10.1007/978-0-387-39940-9_5050.
[53] T. Qin, T.-Y. Liu, J. Xu, H. Li, LETOR: A benchmark collection for research on learning to rank for information retrieval, Inf. Retr. 13 (4) (2010) 346–374.
[54] K. Järvelin, J. Kekäläinen, Cumulated gain-based evaluation of IR techniques, ACM Trans. Inf. Syst. 20 (4) (2002) 422–446.
[55] N. Craswell, Precision at n, in: Encyclopedia of Database Systems, Springer US, Boston, MA, 2009, pp. 2127–2128, http://dx.doi.org/10.1007/978-0-387-39940-9_484.
[56] P.R. Christopher D. Manning, H. Schütze, Introduction to Information Retrieval, Cambridge University Press, 2008, http://dx.doi.org/10.1017/CBO9780511809071.
[57] E. Zhang, Y. Zhang, Average precision, in: Encyclopedia of Database Systems, Springer US, Boston, MA, 2009, pp. 192–193, http://dx.doi.org/10.1007/978-0-387-39940-9_482.
[58] S.M. Beitzel, E.C. Jensen, O. Frieder, MAP, in: Encyclopedia of Database Systems, Springer US, Boston, MA, 2009, pp. 1691–1692, http://dx.doi.org/10.1007/978-0-387-39940-9_492.
[59] W. Shen, J.-Y. Nie, Is concept mapping useful for biomedical information retrieval? in: International Conference of the Cross-Language Evaluation Forum for European Languages, Springer, 2015, pp. 281–286.
[60] A. Das, A. Dasgupta, R. Kumar, Selecting diverse features via spectral regularization, Adv. Neural Inf. Process. Syst. 25 (2012) 1583–1591.