IR Evaluation and Learning in the Presence of Forbidden Documents

David Carmel
dacarmel@amazon.com
Amazon, Israel

Nachshon Cohen
nachshon@amazon.com
Amazon, Israel

Amir Ingber*
ingber@pinecone.io
Pinecone Systems, Israel

Elad Kravi
ekravi@amazon.com
Amazon, Israel

Abstract

Many IR collections contain forbidden documents (F-docs), i.e. documents that should not be retrieved to the searcher. In an ideal scenario F-docs are clearly flagged, hence the ranker can filter them out, guaranteeing that no F-doc will be exposed. However, in real-world scenarios, filtering algorithms are prone to errors. Therefore, an IR evaluation system should also measure filtering quality in addition to ranking quality. Typically, filtering is considered as a classification task and is evaluated independently of the ranking quality. However, due to the mutual affinity between the two, it is desirable to evaluate ranking quality while filtering decisions are being made. In this work we propose $nDCG_f$, a novel extension of the $nDCG_{min}$ metric [14], which measures both ranking and filtering quality of the search results. We show both theoretically and empirically that while $nDCG_{min}$ is not suitable for the simultaneous ranking and filtering task, $nDCG_f$ is a reliable metric in this case.

We experiment with three datasets for which ranking and filtering are both required. In the *PR* dataset our task is to rank product reviews while filtering those marked as spam. Similarly, in the CQA dataset our task is to rank a list of human answers per question while filtering bad answers. We also experiment with the TREC web-track datasets, where F-docs are explicitly labeled, sorting participant runs according to their ranking and filtering quality, demonstrating the stability, sensitivity, and reliability of $nDCG_f$ for this task. We propose a learning to rank and filter (LTRF) framework that is specifically designed to optimize $nDCG_f$, by learning a ranking model and optimizing a filtering threshold used for discarding documents with lower scores. We experiment with several loss functions demonstrating their success in learning an effective LTRF model for the simultaneous learning and filtering task.

CCS Concepts

• **Information systems → Evaluation of retrieval results**; **Document filtering**.

*This work was done while Amir was at Amazon.

SIGIR '22, July 11–15, 2022, Madrid, Spain

ACM ISBN 978-1-4503-8732-3/22/07…$15.00
https://doi.org/10.1145/3477495.3532006

Keywords

Forbidden documents, Ranking, Filtering

ACM Reference Format:
David Carmel, Nachshon Cohen, Amir Ingber, and Elad Kravi. 2022. IR Evaluation and Learning in the Presence of Forbidden Documents. In *Proceedings of the 45th International ACM SIGIR Conference onResearch and Development in Information Retrieval (SIGIR '22), July 11–15, 2022,Madrid, Spain.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3477495.3532006

1 Introduction

One of the main tasks of an Information Retrieval (IR) system is to satisfy its users by ranking relevant content to their requests, typically expressed as vague and imprecise queries. In many cases, the document collection contains forbidden documents, i.e. documents that should not be retrieved to the searcher. Examples include (1) totally forbidden documents for all (offensive, spam), (2) forbidden for a specific use-case (e.g. adult content for children, classified documents for non-certified users). We define such documents as forbidden docs, or F-docs for short.

Typically, IR systems assume an ideal scenario where all F-docs are clearly flagged, hence, a ranker can filter out such documents (and only them) during the retrieval process, guaranteeing that no F-doc will be exposed to the searcher. However, in a real-world scenario, when documents are partially flagged, or non-flagged at all, the filtering algorithm is prone to two types of errors: 1) false-positive (i.e. filtering non F-docs), or 2) false-negative (i.e. leaving F-docs in the list). For example, consider the task of product reviews ranking, where our goal is to rank the reviews according to their helpfulness while filtering forbidden (spam) ones. When filtering is applied by a spam classifier which is prone to errors, the amount of F-docs left in the ranked list, as well as their ranking positions, heavily affects the ranking score as well as the filtering score. Similarly, filtering non F-docs affects both metrics [12].

Therefore, due to the mutual affinity between ranking and filtering, it is desirable to evaluate the system's ranking quality while filtering decisions are being made. We look for an evaluation metric that reinforces a ranker according to three desired criteria: (1) prunes as many F-docs from the retrieved list, (2) does not prune non F-docs from the list, (3) ranks remaining docs according to their relevance to the query while pushing F-docs down the list. We assume an evaluation mechanism that supports graded relevance judgment labels, where the document's relevance is labeled on a multi-level relevance scale. Similarly, a forbidden document is labeled on a multi-level forbidness scale. The *Normalized Discounted*

Cumulative Gain (*nDCG*) [16], a widely used IR metric for handling multi-graded relevance labels, is the first choice to be applied in this case; relevance labels are associated with multi-graded positive scores, the non-relevance label with a zero score[1], and the forbidness labels with multi-graded negative scores. The *nDCG* score for a ranked list containing *F*-docs will reflect their amount, their relative positions in the rank, and their forbidness level.

In this work we adapt the *nDCG* approach, with negatively scored labels for *F*-docs, to evaluate ranking and filtering together. However, as we further discuss in Section 2, *nDCG* is unbounded and unstable in the presence of negatively scored labels [14]. This is one of the reasons why many IR evaluation systems map negative labels to a zero score, thus treating them as non-relevant (e.g. [8]). Instead, we propose $nDCG_f$, an evaluation metric for measuring both ranking and filtering quality of the search results. $nDCG_f$ extends $nDCG_{min}$ [14], a modification of *nDCG* which solves its unboundedness in the presence of negatively scored labels. However, we show that $nDCG_{min}$ is unbounded when ranking and filtering are carried out together, i.e., when the ranker is allowed to retrieve (and rank) a sublist of the search results. $nDCG_f$ solves this flaw by modifying the normalization scheme applied by $nDCG_{min}$ to handle sublist retrieval appropriately.

We experiment with the TREC web-track runs[2] in years 2010-2014 [8–11, 27] demonstrating the inconsistency between $nDCG_{min}$ and $nDCG_f$ when filtering is applied, and that $nDCG_f$ is a reliable [17], stable [6], and sensitive [25] evaluation metric for the simultaneous ranking and filtering task. We then demonstrate how to optimize $nDCG_f$ over two datasets. In the *PR* dataset our task is to rank product reviews while filtering those marked as spam. Similarly, in the CQA dataset our task is to rank a list of human answers per question while filtering bad answers. In both settings there are many *F*-docs (i.e. spam reviews, bad answers) which should not be exposed to the searcher. We propose the *Learning to Rank and Filter* (LTRF) framework, where the task is to rank the good documents in tandem to filtering bad ones. We train ranking and filtering models (*f-rankers*) to optimally rank and filter by additionally tuning a filtering threshold on the model scores. We demonstrate that ranking only, or filtering only, fail to provide a result list with high quality, as manifested by the low $nDCG_f$ score. We then experiment with several loss functions for LTRF, demonstrating their ability to learn a reliable ranking and filtering model that performs better than some baselines.

To summarize, the main contributions of our work are:

- We propose a new evaluation paradigm where f-rankers are evaluated by their ability to rank and filter the search results. We propose $nDCG_f$, a novel evaluation metric that considers both ranking and filtering quality. We demonstrate that $nDCG_f$ is a reliable, stable, and sensitive metric, and is more reliable than $nDCG_{min}$ when filtering and ranking are both applied.
- We present two real-world ranking and filtering tasks where filtering *F*-docs is as important as ranking. We propose an LTRF framework, and several loss functions optimized for $nDCG_f$, demonstrating their effectiveness in learning f-rankers for this task.
- We release a unique large-scale answer ranking and filtering dataset where human answers per question are labeled according to their quality on a multi-level scale, including bad answers labeling. This benchmark can be further used by the community for research on IR evaluation and learning in the presence of forbidden documents.

[1]Typically, non-relevant documents are labeled by a zero score, hence are deemed indifferent by the evaluation system. Recent research [2] shows that different types of non-relevant material affect user satisfaction differently, therefore, a more delicate labeling scheme for non-relevant documents can enhance evaluation effectiveness.
[2]A TREC track's run is a list of ranked results for the track's queries (topics), as provided by one of the track's participants. Qrels is the pull of all judged documents per topic.

2 Related Work

The negative impact of *F*-docs on the user experience has long been recognized by the IR research community. *F*-docs are usually marked by an independent classifier and then filtered out at retrieval time [12]. In this approach filtering accuracy is evaluated independently of the ranking quality. Marking *F*-docs with negative labels allows both tasks to be evaluated by one metric, as was done in the TREC's web-track benchmarks [8–11, 27]. However, TREC participants were not encouraged to filter out the *F*-docs, since the web-track official evaluation metric neglected negative labels by mapping them to a zero score, i.e., treating forbidden and non-relevant documents the same. In the following we cover existing approaches for IR evaluation in the presence of negative labels and some other related works.

Background. We are given a query q, a set of documents $R = \{d_1, \ldots, d_n\}$, a positive integer k, and a set of labels $L = \{l_1, \ldots, l_m\}$. Each label is mapped to a label gain score, $ls : L \rightarrow \mathbb{R}$; $ls(l) \geq 0$ indicates relevance level, and $ls(l) < 0$ indicates forbidness level.

Each document $d \in R$ is labeled by $l(d) \in L$. The document labels are hidden to rankers who are challenged to rank R in "an optimal manner", with respect to the document labels. Let $\pi(R) = (d_1, \ldots, d_n)$, a permutation over R, be the ranker's ranking output. The evaluation task is to measure the "quality" of the top-k documents in the ranked list $\pi(R)$.

nDCG. The Discounted Cumulative Gain (DCG) [16] sums the discounted gain scores of the top-k results in $\pi(R)$:

$$DCG@k(\pi(R)) = \sum_{i=1}^{\min(k,|R|)} \frac{ls(l(d_i))}{\log_2(i+1)}.$$

In order to compare the rankers' performance over a test-set of queries we normalize the DCG value by the DCG score of an ideal ranked list, R^I, generated by sorting R in decreasing order according to the document label gain scores. We define $I_k(R) = \text{DCG@}k(R^I)$. *nDCG* is then normalized by $I_k(R)$:

$$nDCG@k(\pi(R)) = \begin{cases} \frac{DCG@k(\pi(R))}{I_k(R)} & I_k(R) \neq 0 \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

$nDCG@k(\pi(R))$ can then be interpreted as a similarity measure between $\pi(R)$, the ranker output, and R^I, the optimal rank.

$nDCG_{min}$. It is easy to see that when label gain scores are non-negative, *nDCG* is bounded, i.e., $\forall \pi(R), 0 \leq NDCG@k(\pi(R)) \leq 1$. However, in the presence of negative label scores, *nDCG* becomes

unbounded [14]. It can fall below 0, when the DCG value is negative, or even exceeds 1 if both the DCG and the ideal value are negative. Boundedness is essential for reliable statistics computations over the query test set, such as mean and variance [19, 22].

$nDCG_{min}$ [14] tackles the *nDCG* unboundedness problem, in the presence of negatively scored labels. In addition to the computation $I_k(R)$, it also calculates the DCG value of the worst ranked list, R^W, generated by sorting R in increasing order according to the document label scores. We define $W_k(R) = \text{DCG@}k(R^W)$. $nDCG_{min}$ then normalizes the DCG value using min-max normalization with $W_k(R)$ and $I_k(R)$ values:

$$nDCG_{\min}@k(\pi(R)) = \begin{cases} \frac{\text{DCG@k}(\pi(R)) - W_k(R)}{I_k(R) - W_k(R)} & I_k(R) > W_k(R) \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

By applying min-max normalization, using the minimum and maximum DCG values over all permutations of R, boundedness is now explicitly ensured in the presence of negatively scored labels.

Evaluating the Metrics Quality. Moffat [22] identified seven numeric properties that are desirable for evaluation metrics, including boundedness which appears at the top of the list. He showed that *nDCG* satisfies four of these properties, including boundedness, for the case of non-negative label scores. Gienapp et al. [14] showed that the usage of negatively scored labels results in a violation of the boundedness property, while $nDCG_{min}$ preserves all properties satisfied by *nDCG*, including boundedness, in the presence of negatively scored labels.

IR evaluation metrics are also assessed based on three criteria: Stability [6], Sensitivity [25], and Reliability [4]. The stability of an evaluation measure denotes its dependency on the number of queries it is calculated on, and is measured by the error rate associated with the decision about which of two tested systems is better, while varying the query set size [6]. The sensitivity, measured by a bootstrap hypothesis test [25], denotes the metric's ability to successfully tell two systems apart, given that their true performance difference is significant. The reliability of a metric denotes its ability to reflect the actual difference in performance among systems, minimizing deviation from "true" performance rating [4]. It can be interpreted as an approximation to the squared correlation between the relative mean scores observed over the given query test-set and the relative mean scores that would be observed if an infinite number of queries was available. Bodoff and Li [4] applied Generalization theory to evaluate the system reliability, estimating the variance in evaluation results that stems from actual performance difference. Kanoulas and Aslam [17] applied the same analysis for investigation of optimal gains and discount functions for *nDCG*. In Section 4 we describe an experimental study demonstrating the superiority of $nDCG_f$ over *nDCG* with respect to stability, sensitivity, and reliability, and over $nDCG_{min}$ with respect to reliability, in the presence of negatively scored labels and when both ranking and filtering are applied on the search results.

Filtering and Ranking Evaluation. The ranking quality of the retrieved list, and the amount of filtering, are frequently measured independently. Godin et al. [15] described a question answering system that learns when not to answer. They proposed to measure the fraction of questions the system decides not to answer (Answering Rate) and the fraction of questions answered correctly (Precision), consolidating these two metrics by taking the harmonic mean as the final score. Lioma et al. [18] also dealt with the integration of several evaluation measures of retrieved list. They argued about the effectiveness of integrating both relevance and credibility of retrieval results. They presented several evaluation measures that are designed to integrate these metrics together. Maistro et al. [21] suggested a theoretically principled multi-aspect evaluation method that can be used to integrate any number and any type of metrics.

Liu et al. [19] introduced the *Quit While Ahead* (QWA) evaluation approach that captures whether a ranking algorithm filters out unhelpful results. QWA adds a pseudo terminal document, d_t, at the end of the ranked list $\pi(R)$, with relevance label of 1 if no relevant document exists in R, or $\frac{1}{r}\sum_{i=1}^{|R|} ls(l(d_i))$ where r is the number of relevant documents in R. It then evaluates the extended list with a standard IR measure such as $nDCG@(|R|+1)$. Consequently, systems are encouraged to retrieve many relevant docs (increasing the label score of d_t) while reducing the number of non-relevant docs in the list (pushing d_t up in the rank). Recently, Zhang et al. [31] applied this evaluation approach to measure system capability of rejecting unreliable customer reviews retrieved for serving a product question answering system.

The QWA approach assumes binary labels and that $\forall l, ls(l(d)) \geq 0$. While not originally planned for handling negative labels, it can be modified to be used in this case. However it suffers from several drawbacks. Assuming *nDCG* is the metric applied over the list extended with terminal doc, it suffers from unboundedness as *nDCG* does; it can't be used to measure the quality of top-k results as it is only defined for the entire list; and it cannot support multi-graded labels since undesirably, systems may benefit from pruning positively (low-graded) scored documents from the result list, in favor of pushing the virtual document up the list. In the following we describe our proposed metric, $nDCG_f$, a modification of $nDCG_{min}$ which encourages partial retrieval, as *QWA* does, while also preserving boundedness in the presence of negatively scored labels, as $nDCG_{min}$ does.

3 Evaluation of Partial Ranked Lists

Assume that the ranker is allowed to return a ranked sublist $R_f \subseteq R$, by filtering some documents from R (we call such a ranker an f-ranker). In this case, the f-ranker's task is to select a good sublist, out of all possible sublists of R, and the evaluation task is to evaluate the sublist quality. $nDCG_{min}$ successfully evaluates the ranking quality of the entire list R in the presence of negative labels. However, next we show that $nDCG_{min}$ is unbounded when partial retrieval is allowed.

Proposition 3.1. *$nDCG_{min}$ is unbounded in the presence of negatively scored labels, and when partial retrieval is allowed.*

Proof. We prove by example. Lets $R = \{d_1, d_2\}$ with $ls(l(d_1)) = -1$ and $ls(l(d_2)) = +2$. Therefore, $W = W_2(R) = -1 + 2/\log_2(3)) = 0.262$ and $I = I_2(R) = 2 - 1/\log_2(3)) = 1.37$.

For $R_f = (d_1)$, $DCG@2(R_f) = -1$ which is smaller than W. For $R_f = (d_2)$, $DCG@2(R_f) = 2$ which is larger than I. For $R_f = \emptyset$,

$DCG@2(R_f) = 0$ which also deviates from the boundaries. Hence, $nDCG_{min}$ is unbounded for all proper sub-lists of R. □

We also note that when the results list contains a single document $R = \{d\}$, with $ls(l(d)) \neq 0$, and when partial retrieval is allowed, then the ranker task is reduced to a filtering task. There are two desirable outputs, $\emptyset$ if $ls(l(d)) < 0$, or (d) if $ls(l(d)) > 0$. However, $nDCG_{min}((d)) = nDCG_{min}(\emptyset) = 0$ and it turns out that $nDCG_{min}$ cannot differentiate between these two cases, although one of them is preferable depending on d's label score[3]. Therefore, we conclude that $nDCG_{min}$ is not suitable for evaluation when R is a singleton and partial retrieval is allowed.

3.1 $nDCG_f$

To tackle the unboundedness problem of $nDCG_{min}$, in the presence of partial retrieval, we propose $nDCG_f$ which measures the quality of partial ranked lists, i.e. lists constructed by ranking and filtering together. By identifying the ideal and the worst *sublists* over all sublists of R, we can assure an appropriate normalization. Since the DCG value is the (discounted) sum of label scores, it is easy to see that the ideal sublist of R is a sublist of R^I which contains all documents with a non-negative label score (the sublist obtained after filtering out all F-docs from R^I). We mark this sublist by R^I_f, and $I_{k,f}(R) = DCG@k(R^I_f)$. Similarly, the worst sublist of R is a sublist of R^W which contains all documents with a non-positive label score (the sublist obtained after filtering all positively scored docs from R^W). We mark this sublist by R^W_f, and $W_{k,f}(R) = DCG@k(R^W_f)$. It follows directly that $W_{k,f}(R) \leq W_k(R)$ and $I_{k,f}(R) \geq I_k(R)$, therefore,

$$[W_k(R), I_k(R)] \subseteq [W_{k,f}(R), I_{k,f}(R)].$$

Similarly to $nDCG_{min}$, we normalize $nDCG_f$ for any sublist R_f by min-max normalization using the $W_{k,f}(R)$ and$W_{k,f}(R)$ values:

$$nDCG_f@k(R_f) = \begin{cases} \frac{DCG@k(R_f) - W_{k,f}(R)}{I_{k,f}(R) - W_{k,f}(R)} & I_{k,f}(R) > W_{k,f}(R) \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

3.2 Properties of $nDCG_f$

We now prove several properties of $nDCG_f$.

Proposition 3.2. *When the document set R does not contain any F-doc then $nDCG_f$ is equivalent to nDCG.*

Proof. If there are no F-docs in R, then for any $k > 0$, $W_{k,f}(R) = 0$ and $I_{k,f}(R) = I_k(R)$. Therefore, for any $R_f \subseteq R$, $nDCG_f@k(R_f) = nDCG@k(R_f)$. □

Proposition 3.3. *$nDCG_f$ is bounded in the presence of negative labels and when partial retrieval is allowed.*

Proof. For any $k > 0$, $W_{k,f}(R)$ and $I_{k,f}(R)$ are lower and upper bounds on the DCG value of any sublist $R_f \subseteq R$. Therefore, $NDCG_f(R_f, k)$ is bounded ($\in [0, 1]$), as a result of the min-max normalization scheme. □

[3]Note that this also holds for the general case when R contains many documents, all having the same non-zero scored label.

Previously, we argued that $nDCG_{min}$ is not suitable when the ranking task is reduced to filtering (when $R = \{a\}$). We now show that $nDCG_f$ can be properly applied in this case.

Proposition 3.4. *When R contains a single document d with $ls(l(d)) \neq 0$, $nDCG_f$ is reduced to a filtering accuracy metric.*

Proof. Let $R = \{d\}$, and $ls(l(d)) = s \neq 0$. If $s > 0$, then $W_{1,f}(R) = 0$, and $I_{1,f}(R) = s$. There are two possible sublists of R to be retrieved — $\emptyset$, and (d). The DCG values of these two lists are 0 or s, and the $nDCG_f$ values are 0 or 1, respectively, in equivalence to the filtering accuracy score. Similarly, if $s < 0$, $W_{1,f}(R) = s$, and $I_{1,f}(R) = 0$. The DCG values of the two sublists are 0 or s, respectively, and $nDCG_f(\emptyset) = 1$ while $nDCG_f((d)) = 0$, in equivalence to the filtering accuracy score. □

It is also interesting to analyze the score of empty retrieval for a given query with result set R, $nDCG_f@k(\emptyset) = \frac{-W_k(R)}{I_{k,f}(R) - W_{k,f}(R)}$. It can be considered as a lower bound on the desired performance for the given query; bad content gains dominance over good content in any sublist scored below this bound, in which the empty list is preferred. Therefore, this lower bound can be used as a measure of the query's "filtering difficulty". The higher this bound, the more bad content in R which requires intensive filtering efforts.

3.3 $nDCG_f$ vs $nDCG_{min}$

$nDCG_f$ is bounded in the presence of partial retrieval in contrast to the unboundedness of $nDCG_{min}$. We now analyze the discrepancy between the two metrics derived from this property.

Proposition 3.5. *Let R be the result set for a given query. Let $k > 0$, and assume the non-trivial case, i.e. $I_{k,f}(R) > W_{k,f}(R)$. Then, the ranking of IR systems for the given query, based on their $nDCG_f$ scores, is the same as their ranking based on their $nDCG_{min}$ scores.*

Proof. It is enough to show that for the non-trivial case $nDCG_{min}$ and $nDCG_f$ agree on the order of any two IR systems pair for the given query. Let R^1_f, R^2_f be the retrieved (possibly filtered) lists of the two systems. If $nDCG_f@k(R^1_f) > nDCG_f@k(R^2_f)$, then $DCG@k(R^1_f) > DCG@k(R^2_f)$. Therefore, $nDCG_{min}@k(R^1_f) > nDCG_{min}@k(R^2_f)$. □

It turns out that the two metrics agree on system performance order for a specific query. However, as we demonstrate in our experiments, when the system performance is averaged over a set of queries, there is a discrepancy between the two orders. The reason is that the more we filter, the higher the possibility for $nDCG_{min}$ to become unbounded, i.e., the DCG value deviates from the bounds. This anomaly may lead to extreme values of $nDCG_{min}$, with disproportionate effect on the average metric score, and to the discrepancy from $nDCG_f$ ranking. In the following we specify the conditions when $nDCG_{min}$ is guaranteed to be a reliable (bounded) score.

Proposition 3.6. *Let R be a result list for a given query, $F \subseteq R$ is the subset of F-docs, and $I \subseteq R$ is the subset of relevant docs. If $k \leq \min(|R - F|, |R - I|)$ then $nDCG_{min}@k(R) = nDCG_f@k(R)$.*

Proof. Since $DCG@k(R)$ is the same for both metrics, the only difference between the two can be derived from the differences between the lower and upper bounds. If $k \leq |R-I|$ then $W_{k,f}(R) = W_k(R)$ because the top-k worst results do not contain any relevant document. Similarly, if $k \leq |R-F|$ then $I_{k,f}(R) = I_k(R)$ because the top-k results do not contain any F-doc. □

The conclusion from this proposition is that since $nDCG_f$ is bounded, so is $nDCG_{min}$ when $k \leq \min(|R-F|, |R-I|)$. This is typical when $k << |R|$. However, when R is small, as in the review and answer ranking tasks that we experiment with, k is close to $|R|$ and therefore $nDCG_{min}$ may fail for some of the queries due to unboundedness. In addition, when k is large and relatively close to |R|, as in total-recall tasks [24] which are typical in the legal and healthcare domains, we can expect unbounded $nDCG_{min}$ values for many of the queries.

4 Experiments with $nDCG_f$

To study the properties of our new evaluation metric, and the impact of filtering on evaluation, we experiment with three datasets where filtering is required in addition to ranking. The first one is a dataset of product reviews where our task is to rank the set of reviews per product according to amount of their helpful votes, while filtering spam ones. The second is the Community Question Answering (CQA) dataset where the task is to rank the set of answers per question while filtering out bad answers. The third one is the TREC web-track datasets in years 2010-2014 where forbidden documents are explicitly labeled [8–11, 27] and the task is to rank the track's participants.

4.1 Datasets

4.1.1 The product reviews (PR) dataset. This dataset consists of 6.4M customer reviews for 660K products (~10 reviews/product on average; the median is 3 reviews per product)[4]. In this dataset, 21% of the reviews are marked as spam by human experts, and some of the reviews are voted as helpful by other customers. Table 1 shows the distribution of the number of helpful votes and the portion of spam labels over the *PR* dataset.

Our task is to rank the reviews for a given product based on their helpful votes, while filtering out spam reviews. This task represents a real use case of simultaneous ranking and filtering where spam reviews are forbidden for retrieval while good reviews (with many helpful votes) should be ranked at the top of the list. We note that ~33% of the products are associated with a single review only, hence reviews ranking is not required for them and the performance of the f-ranker is impacted by filtering accuracy alone.

Table 1: Helpful votes and spam label distribution over the *PR* dataset.

#helpful votes	0	1	2	≥ 3
%reviews	83.16	7.46	3.02	5.98
%spam	25.53	2.47	2.68	2.69

[4]The *PR* dataset is not a representative sample of entire reviews set on the e-commerce website it was extracted from.

4.1.2 The CQA dataset. In the CQA setting, our task is to rank multiple human answers per question based on their quality and relevance, while filtering out answers that are offensive, mislead, or do not address the question, which are forbidden according to the task specifications.

The CQA dataset[5], sampled from the same e-commerce website, consists of 29.7K product questions, associated with 61K answers (~2.1 answers per question on average). Answer quality was annotated on three level scale using crowd-sourcing[6]:

- **Forbidden (F)**: An answer that is offensive, in bad language, or fails to address the question in any way, (e.g. "I don't know").
- **Satisfying (S)**: The answer is valid but is unjustified or unclear. For example, an answer like "yes" or "no" that does not provide further details.
- **Fully satisfying (P)**: An answer that fully answers the question and is justified and clear.

While the majority of answers (75%) satisfy their questions (40.6% annotated as S and 34.4% annotated as P), 25% are forbidden. Out of all questions, 16.8% have no good answer (i.e., all answers are forbidden). About half of the questions are associated with a single answer where ranking is not required and the model performance is impacted by filtering alone. This distribution stresses the need of filtering out forbidden answers: for one-answer questions, filtering is a must, and for multi-answer questions, ranking forbidden answers at the bottom is often insufficient because these bad answers are still visible to the user. Table 2 shows the distribution of number of answers over the CQA dataset.

Table 2: Distribution of number of answers per question over the CQA dataset.

# answers per question	1	2	3+
% questions	52.0	24.1	23.9

4.1.3 Web-track datasets. These datasets consist of the submitted runs to the TREC web-tracks in years 2010-1014, where each run provides up to 10,000 ranked search results for each of the year's topics. The topic's Qrels holds the aggregated pool of manually judged top-k results for each of the runs, where $k \leq 25$ in all tracks. Table 3 presents the number of runs, the number of topics per year, and the label distribution in the Qrels. In our framework, a document labeled by -2 is considered forbidden, 0 is non-relevant, and 1-4 are positively graded relevance labels.

Table 3: Number of runs, topics, and Qrels in TREC web-tracks 2010 - 2014, and the label distribution over the Qrels.

				Label					
Year	#topics	#Qrels	#runs	−2	0	1	2	3	4
2010	48	25329	56	5.65%	73.70%	15.90%	4.25%	0.54%	0%
2011	50	19381	62	5.25%	78.45%	10.50%	3.67%	2.11%	0%
2012	50	16055	48	5.34%	72.71%	13.75%	2.52%	0.32%	5.34%
2013	50	14474	61	1.62%	69.80%	21.03%	6.36%	1.24%	0.05%
2014	50	14432	44	3.86%	56.90%	26.25%	11.18%	1.59%	0.23%

[5]The dataset is available at https://registry.opendata.aws/ltrf-cqa-dataset/.
[6]We use the Appen crowdsourcing platform for annotation, https://www.appen.com.

Web-track participants were not motivated to filter out negatively labeled results; runs were generated with the goal of optimizing evaluation metrics that are indifferent to negative labels, (TREC web-tracks official evaluation program applies *nDCG* with $ls(l) = \max(0, 2^l - 1)$ [8]; we mark this version by $nDCG_0$). Moreover, since only up to top-25 results per topic were manually judged, many of the run results do not appear in the Qrels, and obviously, many of the Qrels do not appear in the run's results. Therefore, in order to simulate ranking and filtering, we consider each run as an f-ranker of the Qrels; all results that do not appear in the Qrels are omitted, and Qrels not retrieved by the run are considered as filtered results[7]. In this way, the intersection of run results with the Qrels can be considered as the output of a system who applies simultaneous ranking and filtering of the Qrels, and our task is to evaluate these runs according to their ranking and filtering quality. Table 4 exhibits the mean, the standard deviation (std), and the minimum number of run results per topic, after filtering unjudged results, averaged over the web-track's runs.

Table 4: Distribution statistics of the number of results per topic after omitting unjudged results, averaged over TREC web-track runs in years 2010-2014.

	2010	2011	2012	2013	2014
Mean #docs/topic	220.5	171.6	125.9	121.3	141.0
std	57.4	47.8	42.9	59.8	62.5
Min #docs/topic	26.1	39.5	41.6	23.0	44.5

In order to enhance the effect of negative labels, we set the label scores to $ls(-2) = -10$ (severely punishing F-docs), and $\forall l \geq 0, ls(l) = l$.

4.2 $nDCG_{min}$ unboundedness

Our first experiment analyzes the cases when $nDCG_{min}$ fails to provide a bounded score when filtering is being involved. Recall that in the *PR* dataset each product is assigned with a set of reviews, and in the CQA dataset each question is assigned with a set of answers. For both datasets we apply the ordinal loss-based f-ranker (see Section 5) for ranking and filtering, and measure the fraction of queries for which $nDCG_{min}$@3 is unbounded. We call such a case an unbounded query (UBQ). Table 5 shows the percent of unbounded queries, split to over-deviated (DCG@K > I_k) and under-deviated (DCG@k < W_k). In both datasets more than 6% of the queries are unbounded, what makes $nDCG_{min}$ results unreliable. We also note that most of the unbounded queries are over-deviated, probably due to aggressive F-docs filtering by the *ordinal*-based f-ranker. Only 1-2% of the queries are under-deviated, probably due to a few false-positive errors.

We run the same experiment over the web-track datasets, measuring the runs' performance by $nDCG_{min}$@k. Recall that each submitted run, after omitting unjudged results, is considered as an f-ranker of the Qrels set. Table 5 presents the average percent of UBQ per run, in years 2011-2014. For small k values (≤ 50) there were no UBQs at all. However, as Proposition 3.6 states, the larger k is with respect to the result set size, the higher the likelihood for $nDCG_{min}$ to become unbounded. The average Qrels size per topic is 321.7 over the years. Setting k to 300 results in a significant number of unbounded cases. The number of unbounded topics ranges from 9% to 25% over the years, supporting the claim that for these datasets, and for large k values, $nDCG_{min}$ is unreliable when filtering is involved.

[7] According to TREC practices, unjudged results are assumed to be non-relevant (with a zero label), hence the Qrels (virtually) cover the entire document collection. However, assessing a small fraction of submitted results causes many results to not being covered by the Qrels [5]. In this study we ignore the run's unjudged results, evaluating the ranking of the Qrels set only. Qrels documents that do not appear in the results set are considered to have been filtered out by the run.

Table 5: Percent of unbounded queries of the *ordinal*-based f-ranker as measured by $nDCG_{min}$@3 over the *PR* and CQA datasets. For the web-track datasets (2011-2014) we measure the average number of UBQ per run using $nDCG_{min}$@300. Unbounded queries are split to over-deviated (DCG@k > I_k) and under-deviated (DCG@k < W_k).

	PR (k=3)	CQA (k=3)	WT2011 (k=300)	WT2012 (k=300)	WT2013 (k=300)	WT2014 (k=300)
UBQ	15.3%	6.7%	9.2%	11.0%	18.8%	25.3%
Over	14.3%	4.8%	7.2%	10.2%	3.7%	8.3%
Under	1%	1.9%	2.0%	0.8%	15.1%	17.0%

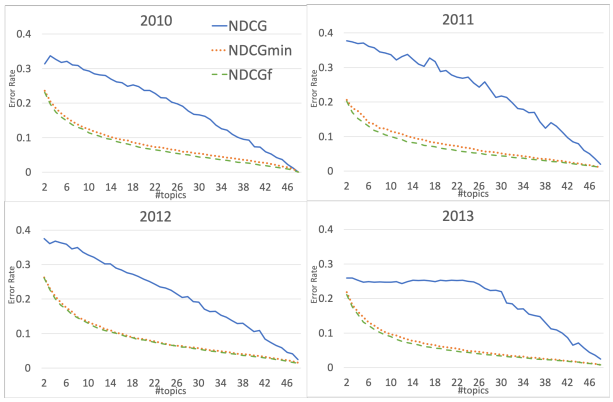

Figure 1: Stability curves of *nDCG*, $nDCG_{min}$, and $nDCG_f$, measured over TREC web-track runs in years 2010-2013

4.3 Stability

The stability of an evaluation measure denotes its dependency on the number of topics it is calculated on [6]. Given a set of n topics and the systems' results, the stability analysis is conducted by repeatedly sampling a random set of i topics, $i \in [2, n]$, evaluating all systems with the selected topics, and measuring the average ratio of erroneous decisions to total decisions across all system pairs (See Equation 1 in [6] for more details). This process is repeated N times for each i value ($N = 200$ in our experiments). Two systems are deemed equal when their absolute score difference does not exceed a fuzziness threshold α (1% in our experiments).

We measure the stability of *nDCG*, $nDCG_{min}$, and $nDCG_f$ over the web-track datasets[8] while setting k to 500. The label scores for DCG

[8] Measuring the metric's stability requires a significant number of rankers to be involved. This does not hold for the *PR* and the CQA datasets, hence we only measure

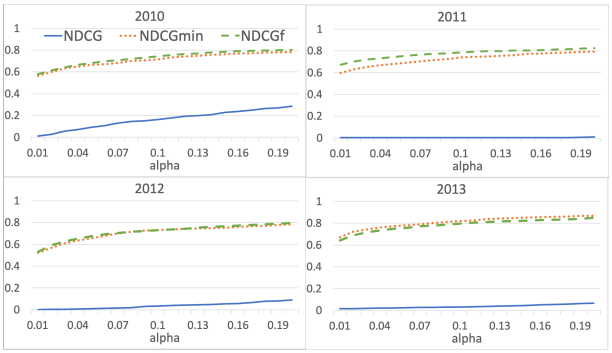

Figure 2: Sensitivity of *nDCG*, $nDCG_{min}$, and $nDCG_f$, measured over TREC web-track runs in years 2010-2013

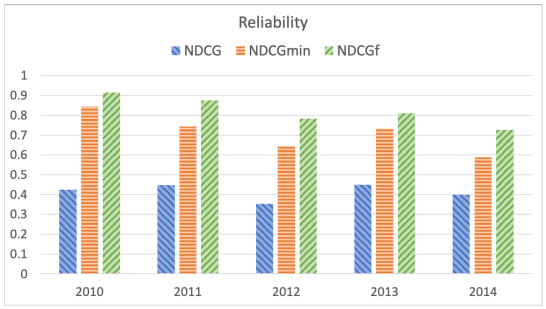

Figure 3: The reliability of *nDCG*, $nDCG_{min}$, and $nDCG_f$, measured over TREC web-track runs

computation of all metrics is set to $ls(-2) = -10$ (reflecting user's intolerance to forbidden results), and $\forall l \geq 0, ls(l) = l$. Figure 1 exhibits the stability (average error rate) curves of the three metrics for the web-track runs in years 2010-2013 (results for 2014 reveal the same behavior). Obviously, the error rate decreases for larger topic sets, however, as we can clearly see, in all years the $nDCG_f$ and $nDCG_{min}$ stability curves dominate the *nDCG* curve, and $nDCG_f$ is as stable as $nDCG_{min}$. The DCG value of all three metrics is exactly the same, and the differences between metrics are only determined by their lower and upper bounds, so we credit the improvement in metric stability to a superior normalization scheme.

4.4 Sensitivity

The sensitivity of an evaluation measure denotes its ability to successfully tell two systems apart, given that their true performance difference is significant. Sakai [25] measured metric sensitivity by comparing the performance of all run pairs using the bootstrap hypothesis test. Given a topic test set Q, for each pair of runs, we randomly sample, with replacement, a set of topics of size $|Q|$ from the test set. We then measure the pair's relative performance on Q and on the sampled set. This pooling process is repeated N times (1000 in our experiments), and the Achieved Significance Level (ASL) measures the portion of cases when the relative performance of the two runs over the two test sets is in disagreement. If $ASL < \alpha$, for small α, we assume that the runs' relative performance holds. The average success ratio over all paired runs ($\#(ASL < \alpha)/\#pairs$) defines the metric sensitivity. The sensitivity curve is a plot of the cumulative success ratio for increasing α values [14].

Figure 2 confronts the sensitivity of *nDCG*, $nDCG_{min}$, and $nDCG_f$, for web-track runs in years 2010-2013. As in all experiments, we set k to 500, $ls(-2) = -10$, and $\forall l \geq 0, ls(l) = l$. For all years, *nDCG* sensitivity is significantly lower than that of $nDCG_{min}$ and $nDCG_f$, for all α values, due to its inferior normalization scheme. $nDCG_f$ is as sensitive as $nDCG_{min}$ (marginally better in 2011).

4.5 Reliability

Reliability is another measure of the metric ability to reflect the actual difference in performance among runs, minimizing deviation from the "true" performance [4, 17]. Given a set of topics and a set of run results, we measure reliability by

$$\Phi = \frac{\sigma^2(run)}{\sigma^2(run) + \frac{\sigma^2(topic)+\sigma^2(run:topic)}{\#topics}}$$

where $\sigma^2(run)$ is the variance in run performance over the topics, $\sigma^2(topic)$ is the variance in topic performance over the runs, and $\sigma^2(run : topic)$ is the run-topic interaction variance [17].

Figure 3 presents the reliability score distribution of *nDCG*, $nDCG_{min}$, and $nDCG_f$, for the web-track runs in years 2010-2014. For all years, the reliability of *nDCG* is significantly inferior to that of $nDCG_{min}$ and $nDCG_f$, due to its improper normalization scheme. Differently from stability and sensitivity, $nDCG_f$ reliability is also significantly superior to that of $nDCG_{min}$.

To conclude, the line of experiments with the web-track runs clearly demonstrates the inferiority of *nDCG* in the presence of negative labels and partial retrieval. The discrepancy between $nDCG_{min}$ and $nDCG_f$ is mostly revealed through the reliability measure, and to some extent through the stability measure. Next we describe how to train an LTRF model that optimizes $nDCG_f$ for ranking and filtering.

5 Learning to rank and filter (LTRF)

In the previous sections we presented the $nDCG_f$ evaluation metric as the natural extension of *nDCG* when filtering is allowed. We now describe how to train an LTRF model that optimizes $nDCG_f$ for the *PR* and CQA settings.

Most LTR approaches [20] learn a ranking model which assigns a single score to each of the documents, with respect to a query, and outputs a sorted document list according to these scores. The ranking model is trained by optimizing a loss function that measures the loss between the ideal ranking and the ranking induced by the model scores. One obstacle that classical LTR approaches have to deal with is optimizing a non-differentiable metric such as *nDCG*, by proposing different surrogate loss functions that are easier to optimize (see, e.g. [23]). Typically, the ranking model is trained by minimizing the loss function using gradient methods (such as stochastic gradient descent for deep learning models, or gradient boosting for tree ensemble models). Similarly to *nDCG*, the new metric, $nDCG_f$, is also non-differentiable, so we rely on similar

stability over the web-track datasets. The same holds for the sensitivity and reliability measures described next.

approaches. In our setting, in addition to the difficulties in optimizing a non-differentiable metric, the f-ranker is also required to filter forbidden results. Learning a threshold filtering value based on score distribution has been studied with the objective of reducing the user overhead in scanning long result lists [3, 30]. Our objective is to reduce bad content in the list by optimizing $nDCG_f$. We experiment with the following approach: we train a ranking model and then optimize a threshold on the model scores to maximize $nDCG_f$ value on the validation set. At inference time we rank the result set by the LTRF model and documents scored below the threshold are filtered out.

5.1 Optimizing $nDCG_f$

The core element of LTR is the loss function to minimize while searching for a ranking model. We now describe the loss functions that we experiment with.

- **Binary cross-entropy (BCE):** we consider the task as a binary classification problem of deciding whether the document should be filtered or not. The model score estimates the likelihood of a document being relevant, i.e. $Pr(ls(l(d)) > 0)$, and is used for document ranking. For this purpose, we define the BCE loss function as

$$\mathcal{L}_{BCE} = \frac{1}{|R|} \sum_{d \in R} BCE\left(s(d), Ind(ls(l(d)) > 0)\right)$$

  where $s(d)$ is the model score for document d, $Ind(\cdot)$ is the indicator function, and $BCE(s, y) = -y \log(s) - (1-y)\log(1-s)$.
- **MSE**: this loss function directly estimates the label gain score for each document. The loss is defined as

$$\mathcal{L}_{MSE} = \frac{1}{|R|} \sum_{d \in R} \left(s(d) - ls(l(d))\right)^2 .$$

- **Ordinal loss** [13]: Given an ordered label set $L = \{l_1, ..., l_m\}$ (where the order is induced by the label score), this model tries to solve $m-1$ binary classification problems: $l(d) \leq l_i$ or $l(d) > l_i$, $i \in [1, m-1]$. The model outputs $m-1$ scores, $s_i(d)$, where $s_i(d)$ approximates $\Pr[l(d) > l_i]$. The loss function is given by:

$$\mathcal{L}_{Ord} = \frac{1}{|R|} \sum_{d \in R} \sum_{i=1}^{m-1} BCE(s_i(d), Ind(l(d) > l_i)).$$

  The final document score is given by the sum $s(d) = \sum_{i=1}^{m-1} s_i(d)$.
- **Weighted ordinal loss:** This loss is similar to that of the ordinal loss, with the following modification. Since the relative importance of the different classification tasks varies according to the difference in label gain scores, we weight the terms in the sum accordingly:

$$\mathcal{L}_{WOL} = \frac{1}{|R|} \sum_{d \in R} \sum_{i=1}^{m-1} w(i) BCE(s_i(d), Ind(l(d) > l_i))$$

  where $w(i) = ls(l_{i+1}) - ls(l_i)$ is the relative weight. The final document score is given by the weighted sum $s(d) = \sum_{i=1}^{m-1} w(i) s_i(d)$.
- **LambdaRank:** this loss function was developed as a surrogate for the non-differentiable *nDCG* metric [7], and is given explicitly in [28], where additional details can be found. We denote this loss function by $\mathcal{L}_{\lambda Rank}$.
- **LambdaRank + BCE:** with this loss function we attempt to learn the ranking model and the filtering threshold together. The loss function is given by

$$\mathcal{L}_{\lambda+BCE} = \mathcal{L}_{\lambda Rank} + \frac{\gamma}{|R|} \sum_{d \in R} BCE(s(d), Ind(ls(l(d)) > 0))$$

  where γ is a hyperparameter controlling the balance between the filtering and ranking parts of the loss.

5.2 LTRF Setting for the Reviews and CQA datasets

Next, we describe the LTRF task for the *PR* and the CQA datasets. In the *PR* setting we are given a product and a list of reviews and our task is to rank the reviews according to their helpful votes while filtering the spam ones. We label the reviews as follow:

$$l(r) = \begin{cases} F & spam(r) \\ 0 & \#votes(r) = 0 \\ 1 & \#votes(r) \in \{1, 2\} \\ 2 & \#votes(r) > 2 \end{cases}$$

In the CQA settings we are given a question with multiple answers, and our task is to rank the answers according to their quality and relevance to the question, while filtering bad one. We label the answers as follows:

$$l(a) = \begin{cases} F & \text{Forbidden}(a) \\ 1 & \text{Satisfying}(a) \\ 2 & \text{Fully Satisfying}(a) \end{cases}$$

In both settings the label scores are set to $ls(F) = -10$ and $ls(l) = l$ for the rest of the labels. In both cases our solution is based on an LTRF model, trained with one of the loss functions described in Subsection 5.1.

We split our datasets to train, validation, and test sets in ratios of (64%,16%,20%) respectively. During training we limit the number of documents (i.e. reviews or answers) per query to at most 3 (via random sampling) to reduce the number of pad documents for queries associated with less than 3 documents. We use DistilBert [26] — a 6-layer distillation of Bert — as our embedding model. The input query is concatenated with each of its associated documents using a special delimiter token and are fed into the embedding model; the output score is computed based on a dense layer that gets as input the CLS token representation[9]. The model's weights (of DistilBERT and of the dense layer) are tuned using the Adam optimizer, while optimizing the loss functions described in Subsection 5.1. Since all losses are listwise functions, we aggregate for each the query its related document scores for computing the gradient for backward propagation. A random hyper parameter tuning was performed for each loss, picking the best combination of parameters on the validation set. The hyper parameters to be optimized include learning rate and batch size. For the $\mathcal{L}_{\lambda+BCE}$ we also tune the γ parameter that weights the two combined losses. Finally, the filtering threshold was selected to maximize $nDCG_f$ on the validation set.

The performance of the LTRF models were examined on the test set. Statistically significant difference in performance was tested using a two-tailed paired t-test. We consider $p < 0.05$ as an indication for a statistically significant difference.

[9]This is the standard usage of the HuggingFace transformers package [1, 29].

5.3 $nDCG_f$ as an accuracy metric

Our first experiment demonstrates $nDCG_f$ ability to be applied as an accuracy metric for single-document queries. We experiment with subsets of such queries – 218K products in the *PR* dataset, and 15K questions in the CQA dataset. For these queries, the task is reduced to filtering only as the f-ranker should decide whether to filter the single document or not.

As discussed in Section 3, $nDCG_{min}$ is not suitable for this task since it is indifferent to the ranker decision. If the f-ranker filters the single document, or if not, it is scored the same. In contrast, $nDCG_f$ score depends on the f-ranker decision and the document's label – 1 for a good decision, i.e., filtering an F-doc or non-filtering a good doc, and 0 for a bad decision.

We experiment with two naive f-rankers; *Filter-all* (who filters the doc), and *Filter-none* (who does not). As expected, the $nDCG_{min}$ scores of both rankers, averaged over all single-answer questions, is zero. Table 6 shows the average $nDCG_f$ scores for both f-rankers over the two datasets.

Table 6: Average $nDCG_f$ scores for two naive f-rankers, measured over a subset of single-document queries.

	Filter-all	*Filter-none*
PR	0.453	0.547
CQA	0.262	0.738

Note that these scores are the accuracy of the two f-rankers (See Prop. 3.4). The score for *Filter-all* is identical to the portion of single-document queries with a negatively labeled document, for which it takes the right decision, while the score for *Filter-none* is the portion of single-document queries with a non-negative labeled document. In both datasets the number of bad documents is lower than the number of good ones, hence it is better to retrieve all than to filter all.

5.4 LTRF Results

The performance of the LTRF models is evaluated over the test set of the two datasets using $nDCG_f$@3, $nDCG_{min}$@3, $nDCG_0$@3, F-docs@3, i.e., the percentage of F-docs remained at the top-3 results after filtering, and the percentage of false-reject errors, i.e., good docs that are filtered out. $nDCG_0$ considers F-docs as non-relevant (zeroing their label score) and therefore measures ranking quality only. We also analyze the percentage of queries left with no documents after filtering as an indication for filtering aggressiveness.

Baselines: We compare the performance of the trained LTRF models with the following baselines:

- *filter-all*: filters all the documents of the query
- *rank-only*: the ordinal-based f-ranker without document filtering
- *filter-only*: the BCE-based f-ranker without ranking. After ranking and filtering is completed by the BCE-based model, remaining documents are shuffled in a random order to undo document ranking

In Table 7 we presents the performance of all models over the two datasets. The following insights can be revealed from the results.

Baselines performance:

- *filter-all*: The $nDCG_f$ score of this f-ranker is the score of the empty list, averaged over all test queries, which can be considered as a lower bound on expected performance (see discussion on "filtering difficulty" in Section 3). As expected, filtering all the documents yields a low $nDCG_f$ score, compared to the trained models, and 100% of good docs filtering. We also note the relatively high score of this f-ranker over the *PR* dataset, as a result of the many spam reviews, and the significant negative label score (-10) — in such a case *filter-all* is preferred over *rank-only*. We also note the negative $nDCG_{min}$ score for the *filter-all* ranker over the CQA dataset, probably due to the large number of unbounded queries of this metric for this ranker.
- *rank-only*: This f-ranker achieves the best $nDCG_0$ scores among all models in the two datasets as this metric is indifferent to forbidden docs, and therefore any filtering policy can only hurt. Albeit, since no F-docs are filtered by this baseline, its $nDCG_f$ score is significantly lower with respect to the trained models. The inferiority in $nDCG_f$ score of *rank-only* with respect to *ordinal*, which it is based on, is of most interest as it is directly attributed to the contribution of the filtering process conducted by *ordinal* on the ranked list.
- *filter-only*: A surprisingly strong baseline, denoting that filtering is crucial in our datasets. $nDCG_f$ score is still behind the trained models, due to the lack of ranking. One of the reasons that this baseline is competitive with the other models, is the large fraction of single document queries for which only filtering is required.

Trained models performance:

All trained models, except *LambdaRank*, are statistically significant better in $nDCG_f$ score than the baselines, over the two datasets. While all trained models perform reasonably well, *ordinal* excels and persistently exhibits high performance, as reflected by high $nDCG_f$ score, low F-docs@3, and the low percentage of good-docs filtering. Its performance is statistically significant better than all other trained models (with the exception of *MSE* in the CQA dataset). Unfortunately, the weighted ordinal loss method, although being theoretically motivated, was unable to improve upon *ordinal* in these settings.

ordinal is also the best f-ranker according to $nDCG_{min}$ over the two datasets. However, the other trained methods are ranked differently according to this metric. For example, in the *PR* dataset, *rank-only* outperforms *filter-all* while the opposite holds according to $nDCG_f$.

Interestingly, in the CQA dataset, *LambdaRank* performs significantly worse than the other loss functions. The reason is that *LambdaRank* is a pairwise loss function which only considers the relative position of the documents in the rank (per query), ignoring their score value. Also, the training set contains a significant portion of single-document queries, while the *LambdaRank* loss is defined on pairs of documents for the same query. Therefore, *LambdaRank* is exposed to a training set that is significantly smaller than the other approaches. When *LambdaRank* is combined with *BCE*, these effects are reduced and the achieved $nDCG_f$ value is competitive with the other trained models.

For the other metrics reported, $nDCG_0$ fails to capture the filtering aspect of the LTRF task as any type of filtering is "punished", hence *rank-only* excels according to this metric. Looking at

Table 7: Performance of LTRF models over the *PR* and the CQA datasets. The best model for each of the metrics is bolded. All trained models (except *LambdaRank* in CQA) show statistically significant gains compared to all baselines with respect to $nDCG_f$@3. *ordinal* is statistically significant better than all other trained models (except *MSE* in the CQA setting).

Dataset	LTRF	$nDCG_f$@3	$nDCG_0$@3	$nDCG_{min}$@3	% F-docs@3	%filtered good docs	%queries left with no docs
PR	*filter-all*	0.6112	0	0.4794	0	100	100
	rank-only (ordinal)	0.5889	**0.3862**	0.4852	25.56	0	0
	filter-only (BCE)	0.763	0.313	0.6392	5.46	19.62	21.63
	BCE	0.7678	0.3021	0.6438	4.9	19.62	21.63
	MSE	0.7729	0.3173	0.6526	4.66	22.74	23.36
	ordinal	**0.7793**	0.3295	**0.6601**	**4.62**	21.7	22.95
	ordinal-weighted	0.7708	0.3148	0.6478	5.19	**18.07**	20.66
	LambdaRank	0.7722	0.34	0.6536	5.91	21.42	22.12
	LamdbaRank+BCE	0.7686	0.3366	0.6494	6.09	22.59	22.61
CQA	*filter-all*	0.3404	0	-0.2723	0	100	100
	rank-only (ordinal)	0.7197	**0.7912**	0.2296	24.49	0	0
	filter-only (BCE)	0.7433	0.7127	0.2251	16.83	8.34	11.31
	BCE	0.7516	0.719	0.2293	16.59	8.34	11.31
	MSE	0.7578	0.7279	0.2466	**16.46**	7.61	10.89
	ordinal	**0.7609**	0.74	**0.2566**	17.10	**6.71**	9.68
	ordinal-weighted	0.7552	0.7311	0.2414	17.13	7.14	10.01
	LambdaRank	0.7045	0.7795	0.2077	25.14	0.03	0.03
	LamdbaRank+BCE	0.7566	0.7323	0.2505	16.97	7.27	10.31

F-docs@3, on the other hand, the trained models achieve a much better score than *rank-only*. Still, in the *PR* dataset, ∼5% of the top-3 results retrieved by the trained models are forbidden, and ∼16% in the CQA dataset. This suggests that there is still a lot of room for improving the filtering process applied by the LTRF models, and this task is still an open challenge and is far from being solved.

6 Conclusion

This work addresses the existence of forbidden documents in data collections, and argues that IR systems should filter retrieved results, in addition to ranking them, to minimize the exposure of such documents to the searchers. We introduced a novel IR evaluation measure, $nDCG_f$, a modification of $nDCG_{min}$ for jointly measuring the ranking and the filtering quality of a retrieved list. We substantiated that in contrast to $nDCG_{min}$, $nDCG_f$ is bounded in the presence negatively scored labels and partial retrieval. We demonstrated $nDCG_f$ for the task of ranking TREC's web-track runs, showing that $nDCG_f$ is more reliable, stable, and sensitive than $nDCG$, and is more reliable than $nDCG_{min}$.

We then introduced the LTRF framework and trained LTRF models to optimally rank the documents, while simultaneously filter forbidden ones, by learning a filtering threshold on the model score. We experimented with the *PR* and CQA datasets, demonstrating that ranking only, or filtering only, fail to provide a ranked and filtered list with high quality, as manifested by the low $nDCG_f$ score. We experimented with several loss functions, showing that ordinal loss excels at this task, significantly outperforming other loss functions.

LTRF is a new research direction that addresses many challenges which deserve further investigation. While our methods succeed to rank and filter, the amount of F-docs left in the retrieved list is still high. We do have some control on the amount of filtering by magnifying the forbidden score. Similarly, by setting the non-relevance label score to an infinitesimal positive or negative value we can control the amount of non-relevant items left in the retrieved list. In general, optimizing the label scores according to the task specifications is a great challenge as it has a huge effect on the quality of both ranking an filtering.

All LTRF models we experimented with are based on learning to rank, and then filter the search results by optimizing a cutoff threshold on the model scores. We experimented with contextual content embedding which considers the query only as the document context. Contextual embedding for answer or review ranking and filtering may benefit from additional context (e.g., answerer reputation score [32]). Further, the current learned threshold is global and query-independent; however, the distribution of document scores is query-specific. Can we learn an optimal threshold for each query? This is another great challenge that we leave for future research.

References

[1] 2020. HuggingFace transformers package. https://huggingface.co/transformers/model_doc/distilbert.html. [Online; accessed 25-May-2021].

[2] Mustafa Abualsaud and Mark Smucker. 2022. The Dark Side of Relevance: The Effect of Non-Relevant Results on Search Behavior. In *ACM SIGIR Conference on Human Information Interaction and Retrieval* (Regensburg, Germany) *(CHIIR '22)*. Association for Computing Machinery, 1–11. https://doi.org/10.1145/3498366.3505770

[3] Avi Arampatzis, Jaap Kamps, and Stephen Robertson. 2009. Where to Stop Reading a Ranked List? Threshold Optimization Using Truncated Score Distributions. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Boston, MA, USA) *(SIGIR '09)*. Association for Computing Machinery, 524–531. https://doi.org/10.1145/1571941.1572031

[4] David Bodoff and Pu Li. 2007. Test Theory for Assessing IR Test Collections. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Amsterdam, The Netherlands) *(SIGIR '07)*. Association for Computing Machinery, 367–374.

[5] Chris Buckley and Ellen M. Voorhees. 2004. Retrieval Evaluation with Incomplete Information. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Sheffield, United Kingdom)

(SIGIR '04). Association for Computing Machinery, 25–32. https://doi.org/10.1145/1008992.1009000

[6] Chris Buckley and Ellen M Voorhees. 2017. Evaluating evaluation measure stability. In *ACM SIGIR Forum*, Vol. 51. ACM New York, NY, USA, 235–242.

[7] Christopher J. C. Burges. 2010. *From RankNet to LambdaRank to LambdaMART: An Overview*. Technical Report. Microsoft Research. http://research.microsoft.com/en-us/um/people/cburges/tech_reports/MSR-TR-2010-82.pdf

[8] Charles L Clarke, Nick Craswell, and Ian M Soboroff. 2010. Overview of the TREC 2010 web track. In *Proceedings of the 19th Text REtrieval Conference*.

[9] Charles L Clarke, Nick Craswell, and Ellen M Voorhees. 2012. Overview of the TREC 2012 web track. In *Proceedings of the 21th Text REtrieval Conference*.

[10] Kevyn Collins-Thompson, Paul Bennett, Fernando Diaz, CLA Clarke, and EM Voorhees. 2013. Overview of the TREC 2013 web track. In *Proceedings of the 22th Text REtrieval Conference*.

[11] Kevyn Collins-Thompson, Craig Macdonald, Paul Bennett, Fernando Diaz, and Ellen M Voorhees. 2014. TREC 2014 web track overview. In *Proceedings of the 23th Text REtrieval Conference*.

[12] Gordon V. Cormack, Mark D. Smucker, and Charles L. Clarke. 2011. Efficient and Effective Spam Filtering and Re-Ranking for Large Web Datasets. *Inf. Retr.* 14, 5 (oct 2011), 441–465. https://doi.org/10.1007/s10791-011-9162-z

[13] Eibe Frank and Mark Hall. 2001. A simple approach to ordinal classification. In *European Conference on Machine Learning*. Springer, 145–156.

[14] Lukas Gienapp, Maik Frobe, Matthias Hagen, and Martin Potthast. 2020. The Impact of Negative Relevance Judgments on NDCG. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management* (Virtual Event, Ireland) *(CIKM '20)*. Association for Computing Machinery, New York, NY, USA, 2037–2040. https://doi.org/10.1145/3340531.3412123

[15] Fréderic Godin, Anjishnu Kumar, and Arpit Mittal. 2019. Learning When Not to Answer: A Ternary Reward Structure for Reinforcement Learning based Question Answering. In *Proceedings of NAACL-HLT*. 122–129.

[16] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)* 20, 4 (2002), 422–446.

[17] Evangelos Kanoulas and Javed A Aslam. 2009. Empirical justification of the gain and discount function for nDCG. In *Proceedings of the 18th ACM conference on Information and knowledge management (CIKM '09)*. 611–620.

[18] Christina Lioma, Jakob Grue Simonsen, and Birger Larsen. 2017. Evaluation Measures for Relevance and Credibility in Ranked Lists. In *Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval* (Amsterdam, The Netherlands) *(ICTIR '17)*. Association for Computing Machinery, New York, NY, USA, 91–98.

[19] Fei Liu, Alistair Moffat, Timothy Baldwin, and Xiuzhen Zhang. 2016. Quit While Ahead: Evaluating Truncated Rankings. In *Proceedings of the 39th International ACM conference on Research and Development in Information Retrieval (SIGIR '16)*. ACM, 953–956.

[20] Tie-Yan Liu. 2011. *Learning to Rank for Information Retrieval*. Springer. I–XVII, 1–285 pages.

[21] Maria Maistro, Lucas Chaves Lima, Jakob Grue Simonsen, and Christina Lioma. 2021. Principled Multi-Aspect Evaluation Measures of Rankings. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 1232–1242.

[22] Alistair Moffat. 2013. Seven numeric properties of effectiveness metrics. In *Asia Information Retrieval Symposium*. Springer, 1–12.

[23] Przemyslaw Pobrotyn, Tomasz Bartczak, Mikolaj Synowiec, Radoslaw Bialobrzeski, and Jaroslaw Bojar. 2020. Context-Aware Learning to Rank with Self-Attention. *ArXiv* abs/2005.10084 (2020).

[24] Adam Roegiest, Gordon V Cormack, Charles LA Clarke, and Maura R Grossman. 2015. TREC 2015 Total Recall Track Overview.. In *TREC*.

[25] Tetsuya Sakai. 2006. Evaluating Evaluation Metrics Based on the Bootstrap. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Seattle, Washington, USA) *(SIGIR '06)*. ACM, 525–532.

[26] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108* (2019).

[27] Ian M Soboroff, Nick Craswell, Charles L Clarke, and Gordon Cormack. 2011. Overview of the TREC 2011 web track. In *Proceedings of the 20th Text REtrieval Conference*.

[28] Xuanhui Wang, Cheng Li, Nadav Golbandi, Michael Bendersky, and Marc Najork. 2018. The lambdaloss framework for ranking metric optimization. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM '18)*. 1313–1322.

[29] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, Online, 38–45. https://www.aclweb.org/anthology/2020.emnlp-demos.6

[30] Chen Wu, Ruqing Zhang, Jiafeng Guo, Yixing Fan, Yanyan Lan, and Xueqi Cheng. 2021. Learning to Truncate Ranked Lists for Information Retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 4453–4461.

[31] Shiwei Zhang, Xiuzhen Zhang, Jey Han Lau, Jeffrey Chan, and Cécile Paris. 2020. Less is More: Rejecting Unreliable Reviews for Product Question Answering. *CoRR* abs/2007.04526 (2020). https://arxiv.org/abs/2007.04526

[32] Zhi-Min Zhou, Man Lan, Zheng-Yu Niu, and Yue Lu. 2012. Exploiting User Profile Information for Answer Ranking in CQA. In *Proceedings of the 21st International Conference on World Wide Web* (Lyon, France) *(WWW '12 Companion)*. Association for Computing Machinery, New York, NY, USA, 767–774. https://doi.org/10.1145/2187980.2188199