Learning to Rank from Relevance Feedback for e-Discovery

Peter Lubell-Doughtie and Katja Hofmann

ISLA, University of Amsterdam,
Amsterdam, Netherlands
peter@helioid.com, k.hofmann@uva.nl

Abstract. In recall-oriented search tasks retrieval systems are privy to a greater amount of user feedback. In this paper we present a novel method of combining relevance feedback with learning to rank. Our experiments use data from the 2010 TREC Legal track to demonstrate that learning to rank can tune relevance feedback to improve result rankings for specific queries, even with limited amounts of user feedback.

1 Introduction

As information retrieval approaches mature, ever more complex tasks are being addressed. One such task is e-discovery, where information concerning a legal argument is uncovered [1]. In contrast to e.g., ad-hoc retrieval, e-discovery is characterized by a much higher user investment and an emphasis on recall. In e-discovery, the cost of not finding a relevant document is much higher than the cost of examining a non-relevant document. Therefore, users usually examine many documents, and are willing to provide feedback in order to increase recall.

In this paper we describe a method that leverages the user feedback provided in an e-discovery task to improve result rankings. We achieve this by combining relevance feedback (RF) and learning to rank on the query level. RF is the most popular method for incorporating information extracted from user interactions into a retrieval system, but a drawback is that its performance can vary substantially depending on parameter settings and the query topic [3]. To address this problem we use learning to rank from RF to automatically tune the amount and nature of RF to a particular query. To do this we construct three types of RF based feature representations, and then apply a pair-wise learning to rank algorithm to select the most promising combination of ranking features.

In our experiments, based on the 2010 TREC Legal track [1], our method outperforms RF, even when only few user interactions are available. The most effective feature set combines retrieval scores from runs with a varying number of expansion terms, and performance gains increase with more user feedback. Our results demonstrate the feasibility of learning to rank for RF per query.

2 Related Work

The most commonly used query expansion method is the Rocchio algorithm [4]. It is based on the vector space representation of queries and documents, and aims to select expansion terms from documents to create a query that can optimally distinguish between relevant and non-relevant documents.

Typically, implementations of the Rocchio algorithm ignore non-relevant documents because excluding terms has been found to be problematic [7]. However, [7] find that negative feedback can be beneficial, especially for difficult queries. Our method avoids the problems of modeling negative feedback by using non-relevant documents in reordering, not query expansion.

Problems with sensitivity to parameter settings have led to substantial research in *adaptive RF*, where parameters such as the number of expansion terms, or the relative weight of expanded and original query terms are tuned using machine learning methods [3, 5]. Previous work considers the task of tuning these settings across queries (e.g., learning to predict the optimal weights for interpolation based on query and result set characteristics). Our work is complementary to these approaches, as it focuses on learning to rank for individual queries.

3 Approach

We model the retrieval task as a series of interactions between retrieval system and user. The user first submits a query and the system returns an initial result list. The user then judges a small number of highly-ranked documents as relevant or non-relevant. These judgments are used for re-ranking using a combination of RF and learning to rank. The re-ranked list is presented to the user, who again judges a small number of documents, and the process continues. Below, we describe our approach for improving document rankings using these judgments.

To facilitate learning that generalizes over documents, we use a feature representation $\phi(d|q)$ that encodes the relationship between the initial query q and a document d. Here, we experiment with three types of features: *term*, *cumulative*, and *constant* features. All feature representations are based on RF given a set of relevance judgments obtained from the user during earlier iterations.

The *term* feature set consists of the frequencies of the top u terms extracted from all judged documents and the top v terms extracted from the complete set of (initially retrieved) candidate documents. We combine these two sets of terms to address problems of sparsity. This term-based representation is similar to learning approaches based on bag-of-words representations. It is expected to be effective if relevant and non-relevant documents can be distinguished based on specific terms used in the documents. However, the feature space is relatively large given the amount of judged training data that can be extracted from user feedback for a single query, which may hurt performance.

The *cumulative* feature set is based on the retrieval scores obtained for a set of expanded queries $(q_0, \ldots, q_i)$. We construct an expanded query for each observed relevant document by obtaining the most frequent terms from that document. We add newly judged documents after each iteration, increasing the number of features with each iteration. This feature representation is based on the observation that some documents are more effective for obtaining RF. We expect it to work well if relevant documents for a given query are well aligned with the documents that are the best sources of expansion terms.

The *constant* feature set is similar to the cumulative feature set in that it is constructed from the retrieval scores of several expanded queries. However, here we do not select expansion terms based on different documents, but we vary the number e of expansion terms extracted from the complete set of documents judged relevant up to the current iteration. Thus, the number of features is constant over iterations, but the terms in the expanded queries that are used to construct the features change over time. As opposed to *cumulative* features, which learn a ranking to align with the best set of documents, *constant* features align with a combination of the best numbers of expansion terms.

After having constructed feature vectors using either of these methods, we apply pairwise learning to rank, which reduces the ranking problem to that of minimizing the number of misclassified document pairs. We solve this classification problem with SGD-SVM [6], and return the documents ordered by their predicted relevance.

4 Experiments

Our experiments are based on the TREC 2010 Legal Track dataset [1], which contains email messages from the Enron corporation lawsuits. The queries (also called *production requests*) are paragraphs describing *relevant* documents and are typically much more complex than a standard ad-hoc query.

Because the kind of interactive experiment we model in our approach is not directly supported by this dataset, we simulate user interactions using the provided queries and relevance judgments. On each iteration we simulate requesting user feedback for 5 relevant and non-relevant documents. To do this we form sample pools of relevant and non-relevant documents by, on each new iteration, adding the top 10 relevant and non-relevant documents as ranked by the **expansion** run (see below) of the previous iteration. We then request user judgments (observed from the judgments provided with the collection) by randomly sampling 5 documents from each pool.

We evaluate the following runs using mean average precision (MAP) and normalized discounted cumulative gain (NDCG) after 25 iterations:

lm - standard language modeling retrieval run using the original query.
expansion - the best-performing setting when using RF alone, $e = 20$.
terms - our approach, with *term* features and $u = 2000$ and $v = 500$.
cumulative - our approach, with *cumulative* features.
constant - our approach, with *constant* features and $e \in \{5, 10, 15, 20\}$.

5 Results and Discussion

Table 1 presents the results for all runs. Our baseline (*lm*) achieved MAP and NDCG scores of 0.067 and 0.565 respectively. The *expansion* run outperforms *lm* with MAP and NDCG scores of 0.0975 and 0.632. Our method using *term* and *cumulative* features also outperforms *lm*, but performs worse than the best RF run (*expansion*), with MAP scores of 0.093 and 0.082, respectively, and NDCG scores of 0.621 and 0.599, respectively.

Best results are achieved with our method and *constant* features. Here, we observe a MAP of 0.109, and an NDCG of 0.645, constituting an improvement of 11.2% in MAP, and of 2% in NDCG over the *expansion* run using RF only. These results demonstrate that our method of combining RF and learning to rank can lead to substantially better performance than using RF alone.

Table 1. Results. Scores are averaged over all queries after 25 iterations.

Metric	**lm**	**expansion**	**terms**	**cumulative**	**constant**
MAP	0.067	0.098	0.093	0.082	**0.109**
NDCG	0.565	0.632	0.621	0.599	**0.645**

Our results indicate that the *constant* feature set is the most appropriate for learning to rank from RF per query. When learning with this feature set, our algorithm finds the most effective combination of expansion runs, with a varying number of expansion terms, for each query. For the *term* feature set, which is most similar to the frequently-used bag-of-words representation, we find that the feature space is too large relative to the small amount of user feedback available when learning per query. Finally, for the *cumulative* feature space we find that selecting the best individual documents for RF does not sufficiently address the problem of noise when selecting terms from small samples of text.

6 Conclusions and Future Work

In this paper we have described a method which uses learning to rank in combination with RF to improve performance beyond that achievable through RF alone. By calculating document scores for multiple retrieval runs using queries expanded with a varying number of high frequency terms, our *constant* approach, we have built useful features for learning to rank. Future research may explore alternative feature representations, combine different features, and investigate per-query learning to rank in the context of other recall-oriented retrieval tasks.

Acknowledgements

This research was partially supported by the Netherlands Organisation for Scientific Research (NWO) under project nr 612.061.814.

References

[1] G. Cormack, M. R. Grossmand, B. Hedin, and D. W. Oard. Overview of the trec 2010 legal track, 2010.
[2] T. Joachims. Optimizing search engines using clickthrough data. In *KDD '02*, pages 133–142, 2002.
[3] Y. Lv and C. Zhai. Adaptive relevance feedback in information retrieval. In *CIKM '09*, pages 255–264, 2009.
[4] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.
[5] G. Pandey and J. Luxenburger. Exploiting session context for information retrieval - a comparative study. In *ECIR*, pages 652–657, 2008.
[6] D. Sculley. Large scale learning to rank. In *NIPS 2009 Workshop on Advances in Ranking*, 2009.
[7] X. Wang, H. Fang, and C. Zhai. A study of methods for negative relevance feedback. In *SIGIR '08*, pages 219–226, 2008.