# Condensed List Relevance Models

Fernando Diaz
Microsoft
fdiaz@microsoft.com

## ABSTRACT

Pseudo-relevance feedback has traditionally been implemented as an expensive re-retrieval of documents from the target corpus. In this work, we demonstrate that, for high precision metrics, re-ranking the original feedback set provides nearly identical performance to re-retrieval with significantly lower latency.

## 1. INTRODUCTION

Pseudo-relevance feedback refers to the use of an initial retrieval to find effective query expansion terms or phrases [1, 5]. The expanded query is often substantially longer than the original query and, as a result, incurs higher latency due to more postings lists being evaluated. This problem is exacerbated by retrieval systems which have been aggressively optimized (both in terms of efficiency and effectiveness) for short, web-like queries. That said, pseudo-relevance feedback consistently improves retrieval effectiveness in many domains [7, 10, 2].

In this paper, we exploit the high overlap between the first and second retrievals in pseudo-relevance feedback in order significantly improve efficiency. This overlap allows fast score computation of those documents already fetched as part of the first stage of the pseudo-relevance feedback process. In order to demonstrate this phenomenon, we computed the overlap between the top ten documents retrieved after pseudo-relevance feedback and those retrieved in the initial retrieval (Table 1). The results show that, with extremely high probability, the top ranked documents after pseudo-relevance feedback have already been fetched. Although this probability falls with the rank of the document in the final retrieval, many retrieval metrics emphasize the top of the ranked list.

We propose the following simple modification to pseudo-relevance feedback: re-rank the initial retrieval instead of re-retrieving. Our experiments test two hypotheses. First, re-ranking the original retrieval set performs as well as re-retrieving. Second, re-ranking the original retrieval is much more efficient than re-retrieving. The evidence from our experiments supports both of these hypotheses.

*ICTIR'15,* September 27–30, Northampton, MA, USA.
© 2015 ACM. ISBN 978-1-4503-3833-2/15/09 ...$15.00.
DOI: http://dx.doi.org/10.1145/2808194.2809491.

**Table 1: Top $n$ locality of pseudo-relevance feedback. We computed the probability of the highest ranked documents *after* pseudo-relevance feedback occurring in the top 1000 documents of the initial retrieval. Probabilities are computed from 1000 random parameter settings for an RM3 model.**

| rank | trec12 | robust | web |
|---|---|---|---|
| 1 | 1 | 0.999 | 0.998 |
| 2 | 0.996 | 0.996 | 0.995 |
| 3 | 0.994 | 0.993 | 0.996 |
| 4 | 0.996 | 0.989 | 0.993 |
| 5 | 0.992 | 0.987 | 0.991 |
| ⋮ | ⋮ | ⋮ | ⋮ |
| 996 | 0.374 | 0.416 | 0.389 |
| 997 | 0.383 | 0.385 | 0.388 |
| 998 | 0.368 | 0.363 | 0.359 |
| 999 | 0.377 | 0.385 | 0.383 |
| 1000 | 0.395 | 0.387 | 0.402 |

## 2. RELATED WORK

Although there has been a great deal of work on improving the effectiveness and robustness of pseudo-relevance feedback techniques, there has been very little work on improving the efficiency of pseudo-relevance feedback.

Cartright *et al.* present several corpus pre-processing methods for improving the efficiency of relevance model-based feedback [4]. The techniques operate under the assumption of massive query expansion where the size of the expanded query is as large as the vocabulary. This allows systems to construct an corpus-level inter-document similarity matrix *a priori* and then exploit this data structure to quickly compute relevance model scores. Although this work is technically exciting, the assumption of a fixed expansion size is problematic. Wu and Fang present a method for incrementally performing pseudo-relevance feedback [12]. This method is similar to ours insofar as the authors reuse computation from the initial retrieval. However, our method seems to be a missing baseline for their experiments.[1] Furthermore, the experiments in Wu and Fang explore very small

[1]Our method is discussed and dismissed as probably leading to poor performance.

numbers of feedback terms (5-20) which limits the generalizability to more realistic, larger expanded queries.[2]

Re-ranking an initial retrieval is not novel. Broder *et al.* present a two-pass method for efficiently scoring documents for standard retrieval [3]. MacDonald *et al.* present a staged ranking method applying fast retrieval methods to establish a candidate set and then applying a slower learning to rank approach on this candidate set [9]. In the context of image retrieval Lin *et al.* proposing re-ranking images using pseudo-relevance information [8]. Finally, Diaz presents a graph-based re-ranking technique which is mathematically related to pseudo-relevance feedback [6].

To the best of our knowledge, we are the first to examine this method for pseudo-relevance feedback on standard text document retrieval.

## 3. ALGORITHMS

Pseudo-relevance feedback techniques often take three parameters: the number of feedback documents ($k$), the number of feedback terms ($m$), and the interpolation weight with the original query ($\lambda$). Another, often overlooked, parameter is the number of final documents retrieved for evaluation ($n$). Standard TREC-style runs often set this value to a default of 1000.

### 3.1 Relevance Model

Relevance modeling refers to pseudo-relevance feedback in the language modeling retrieval framework. Given a query $q$, the maximum likelihood query language model is defined as,

$$p(w|\theta_q) = \frac{\#(w,q)}{\sum_{w' \in q} \#(w',q)} \tag{1}$$

The initial retrieval scores documents according to Kullback-Leibler divergence between the query language model and the document model,

$$\mathrm{D}(\theta_q \| \theta_d) = \sum_{w \in q} p(w|\theta_q) \log \frac{p(w|\theta_q)}{p(w|\theta_d)} \tag{2}$$

where $\theta_d$ is the language model for document $d$. The top $k$ documents for pseudo-relevance feedback are retrieved here.

The expanded query or relevance model (RM) is defined as,

$$p(w|\theta_{\mathrm{RM1}}) = \sum_{i=1}^{k} \frac{p(q|\theta_d)}{\mathcal{Z}} p(w|\theta_d) \tag{3}$$

where $p(q|\theta_d)$, known as the query likelihood, is a simple transformation $\mathrm{D}(\theta_q \| \theta_d)$; and $\mathcal{Z} = \sum_{i=1}^{k} p(q|\theta_d)$. As a heuristic, a system often clips $p(w|\theta_{\mathrm{RM1}})$ to only include the $m$ terms with the largest probabilities. The probability of other terms is set to 0 and the probability of the top $m$ are renormalized.

In practice, linearly interpolating with the original query model (Equation 1) improves performance. This model, RM3, is defined as,

$$p(w|\theta_{\mathrm{RM3}}) = \lambda p(w|\theta_q) + (1-\lambda) p(w|\theta_{\mathrm{RM1}}) \tag{4}$$

where $p(w|\theta_{\mathrm{RM1}})$ is the clipped relevance model. In the second retrieval, documents are again ranked by Equation 2, but this time with $p(w|\theta_{\mathrm{RM3}})$ instead of $p(w|\theta_q)$. At this point, the top $n$ documents are retrieved for evaluation.

### 3.2 Condensed List Relevance Model

We propose the following small change to relevance model retrieval. Instead of retrieving $k$ in our initial retrieval, we retrieve $n$. We still estimate the relevance model with the smaller set of $k$ documents (Equation 3). However, our final ranking of $n$ documents is a re-ranking of the initial retrieval. We refer to this method as the *condensed list relevance model* (CLRM) because we remove unscored documents in the same way condensed list evaluation metrics remove unjudged documents [11]. In practice, we use the interpolated relevance model (Equation 4) and refer to our algorithm as CLRM3.

## 4. METHODS

We evaluate our method on three *ad hoc* datasets: trec12 consists of topics 51-200 associated with the Tipster 1 and 2 disks; robust consists of topics 301-450 and 601-700 associated with the Tipster 4 and 5 disks; web consists of topics 1-200 associated with Clue Web 2009 Category B. We removed documents from web with a Waterloo spam score less than 70.[3] We indexed corpora using Indri with the SMART stopword list and with Krovetz stemming. We ran our experiments on a cluster of nine machines, each with 24 Intel Xeon 2.27GHz CPUs and 24 GB of RAM. All indexes were built and accessed locally with one IndriRunQuery process per query.

We use the Indri implementation of RM3 and implemented our algorithm by adding roughly fifteen lines of code to `IndriRunQuery.cpp`. This patch will be available on acceptance.

We performed ten-fold cross-validation to tune the $k$, $m$, and $\lambda$ parameters. The Dirichlet $\mu$ was fixed at 2500, the Indri default, and $n$ was fixed at 1000, as is customary in TREC evaluations. We ran one cross-validation for each of the presented metrics so the results reflect the performance of an algorithm whose parameters are cross-validated for that metric. We compare methods by concatenating the ten testing folds and then performing a paired Student's $t$-test on queries.

## 5. RESULTS

We present effectiveness results in Table 2 broken down by precision and recall oriented metrics. Because Table 1 suggests that most of the top documents will be preserved with CLRM3, we expect performance for high precision metrics to be comparable to RM3. Indeed, the results in Table 2 confirm this suspicion. Across all conditions, we find that CLRM3 performs as well as RM3; in one case, it outperforms RM3. This provides evidence supporting our first hypothesis. The recall-oriented metrics describe the performance farther down the ranked list. In particular, the recall metric suggests that RM3 is more effective at retrieving more relevant documents, although the effect may not be noticeable at the top of the ranked list.

We inspected the effectiveness of individual query and parameter settings in our sweep. We found almost identical

[2]Previous results demonstrate that optimal parameters for relevance models lie in the 75-100 range for number of terms [6, Table 3].

[3]https://plg.uwaterloo.ca/ gvcormac/clueweb09spam/

**Table 2: Effectiveness results for precision and recall oriented metrics. Numbers in bold indicates statistically significant improvement over the alternative for that metric using a Student's paired $t$-test ($p < 0.05$).**

(a) Precision-oriented metrics

| | RR | NDCG5 | NDCG10 |
|---|---|---|---|
| trec12 | | | |
| RM3 | 0.7343 | 0.6047 | 0.5629 |
| CLRM3 | **0.7599** | 0.6091 | 0.5661 |
| robust | | | |
| RM3 | 0.7100 | 0.4810 | 0.4767 |
| CLRM3 | 0.7100 | 0.4808 | 0.4757 |
| cw09b | | | |
| RM3 | 0.4907 | 0.2161 | 0.2032 |
| CLRM3 | 0.4907 | 0.2161 | 0.2032 |

(b) Recall-oriented metrics

| | MAP | Rprec | recall |
|---|---|---|---|
| trec12 | | | |
| RM3 | **0.3164** | **0.3532** | **0.6834** |
| CLRM3 | 0.2921 | 0.3410 | 0.5939 |
| robust | | | |
| RM3 | **0.2892** | 0.3140 | **0.7625** |
| CLRM3 | 0.2767 | 0.3140 | 0.6914 |
| cw09b | | | |
| RM3 | 0.1613 | 0.2108 | **0.4712** |
| CLRM3 | 0.1606 | 0.2108 | 0.4644 |

performance between RM3 and CLRM3 for high precision metrics (Table 3). This similarity in performance disappears for recall-oriented metrics. In Figure 1, we present scatterplots for those query-parameter samples where performance differed. We notice that recall-oriented metrics exhibit a horizontal banding effect for those queries whose performance can improve when in the RM3 condition but are constrained by the original retrieval for the CLRM3 condition.

In terms of efficiency, CLRM3 is substantially faster than RM3, operating at 5.68% (trec12), 5.03% (robust), and 10.1% (web) of RM3 at optimal parameter settings. In order to address the concern that this improvement was an artifact of our single machine infrastructure, we distributed the web index across our nine servers and reproduced our experiments. The efficiency improvements were identical to those on the single machine architecture. These experiments support our second hypothesis.

The latency incurred by these approaches can be affected by our pseudo-relevance feedback parameters. Here, we define latency as the per-query algorithm runtime above a baseline without feedback (i.e. query likelihood). We present this analysis in Figure 2. In general, we find that changing the number of feedback documents ($k$) does not affect performance for either algorithm. The interpolation weight ($\lambda$) improves the runtime for RM3. This is most

**Table 3: Fraction of runs with equal metric values for RM3 and CLRM3 during a parameter sweep. For a random setting of parameters in our sweep, this is the probability that RM3 and CLRM3 will perform identically.**

| | RR | NDCG5 | NDCG10 | MAP | Rprec | recall |
|---|---|---|---|---|---|---|
| trec12 | 0.958 | 0.989 | 0.976 | 0.087 | 0.699 | 0.129 |
| robust | 0.968 | 0.995 | 0.985 | 0.297 | 0.901 | 0.397 |
| cw09b | 0.894 | 0.998 | 0.992 | 0.403 | 0.947 | 0.562 |

likely a result of term weights becoming more skewed, allowing more aggressive pruning of candidate documents by the retrieval system. Because CLRM3 scores a fixed set of documents, we do not see this effect for the new algorithm. The number of feedback terms ($m$) significantly affects runtime. A naïve linear model predicts each additional term adding $\sim$ 8000 milliseconds to the retrieval for RM3 and $\sim$ 100 milliseconds for CLRM3.

## 6. CONCLUSION

We found our results as impressive as the algorithm was simple. We believe that researchers and practitioners should strongly consider condensed list pseudo-relevance feedback when there are efficiency concerns with two-pass pseudo-relevance feedback.

## 7. REFERENCES

[1] R. Attar and A. S. Fraenkel. Local feedback in full-text retrieval systems. *J. ACM*, 24(3):397–417, July 1977.

[2] M. Bendersky, D. Metzler, and W. B. Croft. Effective query formulation with multiple information sources. In *WSDM*, 2012.

[3] A. Z. Broder, D. Carmel, M. Herscovici, A. Soffer, and J. Zien. Efficient query evaluation using a two-level retrieval process. In *CIKM*, 2003.

[4] M.-A. Cartright, J. Allan, V. Lavrenko, and A. McGregor. Fast query expansion using approximations of relevance models. In *CIKM*, 2010.

[5] W. B. Croft and D. J. Harper. Using probabilistic models of document retrieval without relevance information. *Journal of Documentation*, 35(4):285–295, 1979.

[6] F. Diaz. Regularizing query-based retrieval scores. *Information Retrieval*, 10(6):531–562, December 2007.

[7] V. Lavrenko. *A Generative Theory of Relevance*. PhD thesis, University of Massachusetts, 2004.

[8] W.-H. Lin, R. Jin, and A. Hauptmann. Web image retrieval re-ranking with relevance model. In *Web Intelligence*, 2003.

[9] C. Macdonald, N. Tonellotto, and I. Ounis. Effect of dynamic pruning safety on learning to rank effectiveness. In *SIGIR*, 2012.

[10] D. Metzler and W. B. Croft. Latent concept expansion using markov random fields. In *SIGIR*, 2007.

[11] T. Sakai. Alternatives to bpref. In *SIGIR*, 2007.

[12] H. Wu and H. Fang. An incremental approach to efficient pseudo-relevance feedback. In *SIGIR*, 2013.

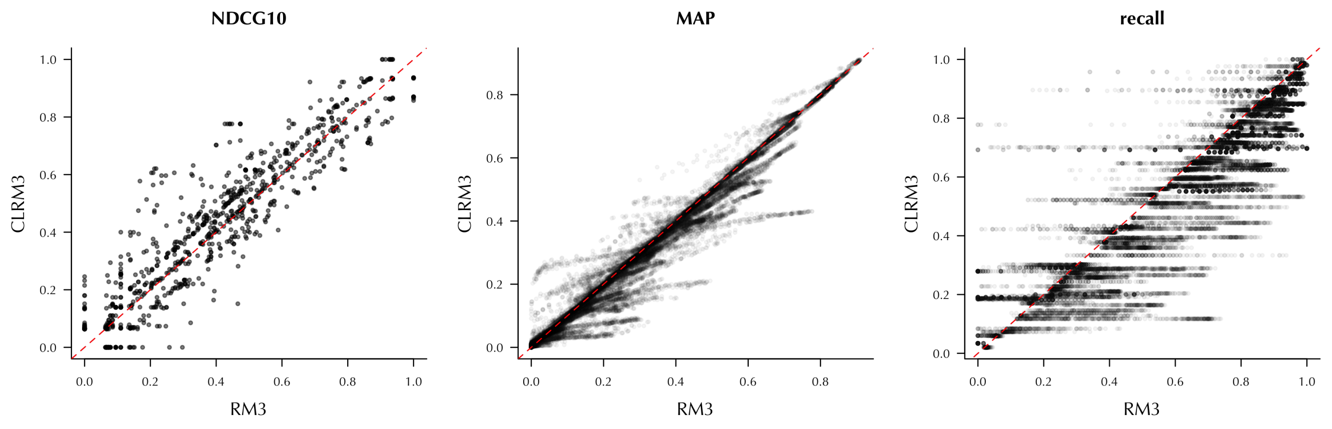

**Figure 1: Correlation between RM3 and CLRM3 for metrics with different recall orientation on the web dataset. Each point represents the performance of each algorithm for a unique query-parameter setting. We have omitted points lying *exactly* on the diagonal (see Table 3). Horizontal banding reflects queries whose performance can increase by retrieving new documents from a second retrieval.**

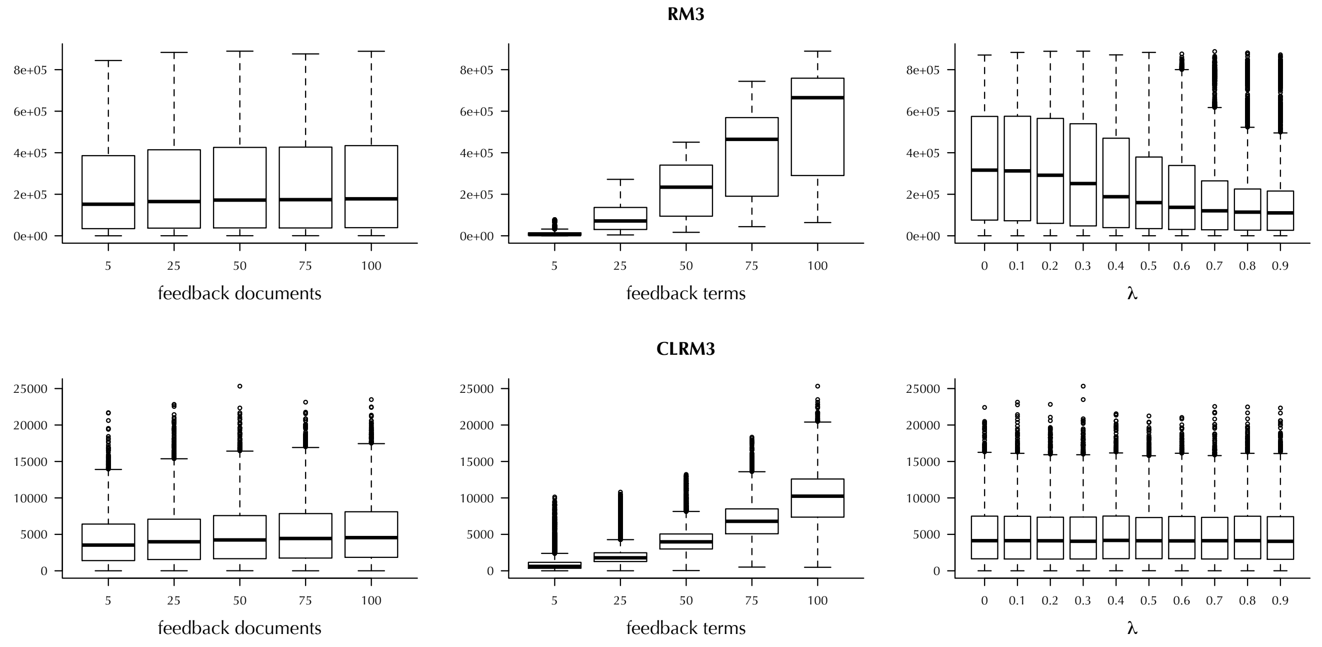

**Figure 2: Distribution of latency (ms) incurred from relevance modeling as a function of relevance model parameters. Note that while the vertical axes are comparable for a given algorithm, the axes for CLRM3 are *significantly* different from those of RM3. Runtime is measured on the full set of parameters in the sweep for all queries in web.**