A Rank-Aggregation Approach to Searching for Optimal Query-Specific Clusters

Oren Kurland and Carmel Domshlak
Faculty of Industrial Engineering and Management
Technion — Israel Institute of Technology
Haifa 32000, Israel
{kurland,dcarmel}@ie.technion.ac.il

ABSTRACT

To improve the precision at the very top ranks of a document list presented in response to a query, researchers suggested to exploit information induced from *clustering* of documents highly ranked by some initial search. We propose a novel model for ranking such (*query-specific*) clusters by the presumed percentage of relevant documents that they contain. The model is based on (i) proposing a palette of "witness" cluster properties that purportedly correlate with this percentage, (ii) devising concrete quantitative measures for these properties, and (iii) ordering the clusters via aggregation of rankings induced by these individual measures. Empirical evaluation shows that our model is consistently more effective than previously suggested methods in detecting clusters containing a high relevant-document percentage. Furthermore, the precision-at-top-ranks performance of this model transcends that of standard document-based retrieval, and competes with that of a state-of-the-art document-based retrieval approach.

Categories and Subject Descriptors: H.3.3 [Information Search and Retrieval]: Retrieval models, Clustering

General Terms: Algorithms, Experimentation

Keywords: clusters, query-specific clustering, cluster properties, optimal cluster, language models, rank aggregation

1. INTRODUCTION

Users of search engines expect to see the documents pertaining to their queries at the very top ranks of the retrieved results. To address this challenge, some researchers proposed to utilize information captured by *clusters* of the documents most highly ranked by some initial search (a.k.a. *query-specific clusters*) [28, 34, 11, 22, 31, 30, 24, 18, 26, 25].

Table 1 illustrates one of the most significant (potential) merits in employing query-specific clustering. The first two rows depict the precision of the top-5 retrieved documents (p@5) performance obtained over three TREC benchmarks [33] by a standard *language model* approach [27] and by a state-of-the-art retrieval approach, namely, the *relevance model* (RM3) [20, 1]. Now, suppose that we use some clustering algorithm (for each query) upon the 50 highest-ranked documents (henceforth referred to as the "initial list") by the standard language model approach to produce clusters of 5 documents; the bottom-most row in Table 1 then shows the (average) percentage of relevant documents in the *optimal cluster* — the cluster that contains the highest such percentage. (Further technical details regarding the retrieval and clustering methods are elaborated in Section 4.) The message rising from Table 1 is clear: if we were able to *automatically detect for each query its optimal cluster(s)*, and then position their constituent documents at the top of the returned results, then the resultant retrieval performance would have been substantially better than that of current (state-of-the-art) document-based retrieval approaches.

	WSJ	TREC8	AP
Standard LM	53.6	50.0	45.7
Relevance Model	58.8	53.6	50.3
Optimal Cluster	**81.5**	**83.6**	**79.6**

Table 1: The average percentage of relevant documents in an optimal cluster of 5 documents in comparison to the p@5 performance obtained by a standard language model (LM) approach and a relevance model (RM3) [1].

The conclusion just drawn has long been echoed by researchers who employed various clustering techniques to top-retrieved documents [11, 31, 14, 25] following van Rijsbergen's cluster hypothesis [32]. Nevertheless, there are relatively few reports on concrete approaches for automatically detecting optimal (query-specific) clusters [34, 24, 18, 26, 25]. Furthermore, most of these approaches are based on comparing a cluster representation with that of the query, and this turns out to have only limited success [34, 24, 26].

In this paper we propose a general framework for tackling the problem of (automatically) identifying optimal query-specific clusters. The intuitive basic principle underlying our proposal is to

- identify a palette of *"witness" properties* of query-specific clusters, such that the satisfaction-degree of each property can be arguably hypothesized to imply about the percentage of relevant documents in a cluster, and
- aggregate the "testimonies" of these "witnesses" to overcome their individual biases and weaknesses, and thereby improve the robustness of the overall cluster-evaluation.

SIGIR'08, July 20–24, 2008, Singapore.

Technically, this evidence-aggregation principle boils down to *rank aggregation* of rankings induced over the clusters by the individual "witness" properties. In particular, here we propose four such qualitative properties that we hypothesize to be connected to the percentage of relevant documents in query-specific clusters. All four properties are derived from insights — most of which are only indirectly related to the task of optimal-cluster detection — on individual (and arbitrary sets of) relevant documents that have been gained in prior work on (i) relevance feedback [29], (ii) cluster-based retrieval [24, 18], and (iii) re-ranking the initial document list using graph-based approaches [17, 7, 18]. We then suggest concrete estimation models for quantifying the extent to which a cluster exhibits each of these qualitative properties, and utilize the aggregation of the induced estimates for ranking the clusters at hand.

Through an array of experiments conducted over TREC corpora we show that our approach is more effective in detecting clusters containing a high relevant-document percentage than previous (state-of-the-art) methods [24, 18]. Furthermore, posting the constituent documents of the cluster most highly ranked by our approach at the top of the retrieved document list yields precision-at-top-ranks performance that is substantially better than that of standard document-based retrieval; the performance also competes with that of a state-of-the-art document-based retrieval approach, namely, the aforementioned relevance model (RM3) [20, 1].

2. RANKING CLUSTERS

2.1 Preliminaries

Let q, d and $\mathcal{D}$ denote a query, a document and a corpus of documents respectively. We use $\mathcal{D}^N_{\text{init}}$ (or simply $\mathcal{D}_{\text{init}}$) to denote the set of N most highly-ranked documents by some initial search performed in response to q. We assume that some clustering algorithm was run on $\mathcal{D}_{\text{init}}$ and produced a set of *document clusters* $Cl(\mathcal{D}_{\text{init}}) = \{c_1, \cdots, c_M\}$, which we wish to rank by the (presumed) percentage of relevant documents that they contain[1].

Our ranking framework utilizes statistical language models [27, 5]. We use $p_x(\cdot)$ to denote a (smoothed) unigram language model induced from x (a query, a document, or a set of documents); language-model induction details are described in Section 4.1. We also make use of Kronecker's delta function: for a predicate s, $\delta[s] = 1$ if s is true, and 0 otherwise.

Our cluster-ranking methods utilize information about the relative positioning of documents in ranked lists. For example, we will be interested in how cluster's constituent documents are ranked when a search is performed upon the entire corpus using a language model induced from the cluster (details below). To that end, we set the following definitions.

Let $\mathcal{S} \subseteq \mathcal{D}$ be a set of documents and $p_x(\cdot)$ be some (unigram) language model. We use the standard language model KL-retrieval approach [19] to rank the documents in $\mathcal{S}$ with respect to $p_x(\cdot)$ by scoring each document $d \in \mathcal{S}$ with

$$KL\left(p_x(\cdot) \,\middle|\middle|\, p_d(\cdot)\right) = \sum_w p_x(w) \log \frac{p_x(w)}{p_d(w)}, \qquad (1)$$

[1]We discuss the computational aspect of employing such online clustering in Section 4.

where the summation is over all terms w in the vocabulary. We also define the schematic function $\mathcal{L}_{p_x(\cdot)||\mathcal{S}}(\cdot) : \{1, \ldots, |\mathcal{S}|\} \rightarrow \mathcal{S}$ to provide a strict ranking of $\mathcal{S}$ that is consistent with the scores of its documents. (Score ties are broken by document IDs.) For any subset $\mathcal{R} \subseteq \mathcal{S}$, we quantify the relative position of $\mathcal{R}$'s documents in the ranking induced by $\mathcal{L}_{p_x(\cdot)||\mathcal{S}}(\cdot)$ over $\mathcal{S}$ using the widely used *non-interpolated average precision* measure with cutoff ν [33]:

$$AP_{p_x(\cdot)||\mathcal{S}}(\mathcal{R}; \nu) \stackrel{def}{=} \frac{1}{|\mathcal{R}|} \sum_{i=1}^{\nu} \frac{\delta\left[\mathcal{L}_{p_x(\cdot)||\mathcal{S}}(i) \in \mathcal{R}\right]}{i} \sum_{j=1}^{i} \delta\left[\mathcal{L}_{p_x(\cdot)||\mathcal{S}}(j) \in \mathcal{R}\right]. \qquad (2)$$

For example, if $\mathcal{R}$ is the set of relevant documents for query q, then $AP_{p_q(\cdot)||\mathcal{S}}(\mathcal{R}; 1000)$ is the (non-interpolated) average precision at 1000 — TREC's widely-used measure — obtained by the standard KL-retrieval approach.

As mentioned above, the underlying principle of our proposed framework for detecting clusters that contain a high percentage of relevant documents is integration of witness-properties via rank aggregation. Informally, the framework is based on

(1) identifying a set of qualitative *properties* of a cluster in $Cl(\mathcal{D}_{\text{init}})$ that are purportedly connected with the percentage of relevant documents the cluster contains,

(2) committing to concrete *cluster-ranking functions* for quantifying the extent to which a cluster exhibits each of these qualitative properties, and

(3) committing to a principle for *aggregating* the rankings induced by these ranking functions over the clusters in $Cl(\mathcal{D}_{\text{init}})$.

In what follows, we describe our cluster-ranking model in terms of these three components.

2.2 Characterizing optimal clusters

We begin with qualitatively describing the four characteristic properties of clusters in $Cl(\mathcal{D}_{\text{init}})$ that we argue to be connected with the percentage of relevant documents that the clusters contain. The qualitative description of each property is then paired with a concrete ranking function that quantitatively measures fulfillment of the property by a cluster.

*Query faithfulness (***QF***).* Analogously to the case of ranking documents in response to a query, one might expect that the more a cluster (as a whole unit) is "similar" to the query the higher are the chances that it contains a high percentage of relevant documents. There is evidence, however, that estimating (different notions) of cluster-query similarity by the "match" between a cluster and a query representation (e.g., their induced language models) is not effective for detecting the desired clusters [34, 24]. Stratifying this estimator, recent work on (re-)ranking the initial list $\mathcal{D}_{\text{init}}$ for obtaining high precision at top ranks using graph-based approaches [17, 7] suggests that a document is likely to be relevant if it is highly similar both to the query and to many documents in $\mathcal{D}_{\text{init}}$ that are highly similar to the query.

Following this observation we assume that a cluster containing a high percentage of relevant documents groups doc-

uments that exhibit high similarity to the query[2]. This hypothesis leads us to take a document-mediated approach to assessing the "faithfulness" (i.e., similarity) of the cluster to the query. Specifically, we score cluster c by the relative positioning of its constituent documents in the ranking induced on the initial list $\mathcal{D}_{\text{init}}$ by the document-query (language-model) similarity, that is,

$$Score_{QF}(c) \stackrel{def}{=} AP_{p_q(\cdot)||\mathcal{D}_{\text{init}}}(c;\nu) + \varepsilon; \quad (3)$$

the smoothing parameter ε, which we set here and after to $\frac{1}{\nu+1}$, ensures that the score of a cluster is greater than zero.

***Self faithfulness* (SF).** Work on relevance feedback showed that inducing a "query model" from a set of relevant documents and using this model for ranking *all* documents in the corpus yields highly effective retrieval performance [29]. Now, suppose that cluster c contains only relevant documents; then, we should expect that ranking the corpus based on a query model induced from c will result in relevant documents being positioned high in the ranked list. While at first view this prediction is not very operational, when combined with our assumption on the content of c it implies that, in particular, the documents forming c will be positioned high in the ranked list. We therefore post as our second witness property the "faithfulness" of the ranking induced on the corpus by the cluster model to the cluster's constituent documents. To quantify the degree of cluster's "self faithfulness", each cluster in $Cl(\mathcal{D}_{\text{init}})$ is scored by

$$Score_{SF}(c) \stackrel{def}{=} AP_{p_c(\cdot)||\mathcal{D}}(c;\nu) + \varepsilon. \quad (4)$$

***Initial-list faithfulness* (ILF).** The initial list $\mathcal{D}_{\text{init}}$ can be thought of as reflecting the "corpus-context" of the query by the virtue of the way it is created. Indeed, this premise is taken by many effective pseudo-feedback-based approaches for automatic query expansion [2, 35]. Furthermore, some recent work has shown that clusters in $Cl(\mathcal{D}_{\text{init}})$ that are similar to many documents in $\mathcal{D}_{\text{init}}$ — and hence can be thought of as reflecting the corpus-context of the query — tend to contain a high percentage of relevant documents [18]. Here, we propose to measure the extent to which a cluster exhibits the query's corpus-context by the (relative) situation of the documents from the initial list $\mathcal{D}_{\text{init}}$ in the ranking induced by the cluster model over the *entire corpus*, that is,

$$Score_{ILF}(c) \stackrel{def}{=} AP_{p_c(\cdot)||\mathcal{D}}(\mathcal{D}_{\text{init}};\nu) + \varepsilon. \quad (5)$$

***Peer faithfulness* (PF).** The clusters in $Cl(\mathcal{D}_{\text{init}})$ are often thought of as "reflecting" different query-related aspects of the corpus that are manifested in the initial list $\mathcal{D}_{\text{init}}$ [28, 34, 11, 22, 24, 18]. Consequently, clusters that are associated with many such aspects (while still being relatively focused by the virtue of being "clusters") potentially contain a high percentage of relevant documents. This hypothesis leads us to employ the "faithfulness" of the cluster to the corpus-specific query-related aspects (as manifested in its *peer* clusters) as yet another witness property of the clusters we look for. To quantitatively assess the "peer-faithfulness" of cluster c, we score it by the positioning of c's constituent documents in *all* rankings induced over the entire corpus by the cluster models of c's peer clusters $Cl(\mathcal{D}_{\text{init}})\backslash\{c\}$, notably

$$Score_{PF}(c) \stackrel{def}{=} \frac{1}{M-1}\sum_{c' \neq c} AP_{p_{c'}(\cdot)||\mathcal{D}}(c;\nu) + \varepsilon. \quad (6)$$

[2]We focus on clustering techniques that do not have any explicit knowledge of the query in hand.

2.3 Rank aggregation

While we hypothesize that clusters containing a high percentage of relevant documents are likely to exhibit the four witness properties presented above, obviously, "likely" does not mean "always". Our suggestion is thus to use these four properties as a set of judges, and compute (via rank-aggregation) a "consensus" cluster-ranking aiming at improving (on average) over each of the individual cluster-rankings induced by the four properties. Note that,

- the "degree of exhibition" of all our four properties is measured in the *same terms* of the average (non-interpolated) precision obtained by some ranking of documents, and thus
- all these cluster scores have somewhat similar nature: all lie in the interval $[0, 1+\varepsilon]$, and each cluster is associated with a vector of such fuzzy scores in $[0, 1+\varepsilon]^4$.

There are a number of reasonable "aggregation functions" from $[0, 1+\varepsilon]^\kappa$ to $[0, 1+\varepsilon]$ that assign a score to a "conjunctive" combination of κ fuzzy-scored properties (e.g., see Zimmermann's textbook [41]). To promote clusters that exhibit a larger number of properties to high extent, here we adopt the algebraic product aggregation function, and score cluster c with respect to the set of properties Π by

$$Score_{\wedge}(c;\Pi) \stackrel{def}{=} \prod_{\mathcal{P}\in\Pi} Score_{\mathcal{P}}(c). \quad (7)$$

Specifically, the **AllProp** algorithm that will be of our focus here, uses all four properties for scoring cluster c, that is,

$$Score_{AllProp}(c) \stackrel{def}{=} Score_{\wedge}(c;\{QF, SF, ILF, PF\}).$$

In Section 4.3 we compare the performance of this aggregation approach to that of some alternatives — regular sum and Borda's method.

3. RELATED WORK

Clustering the top-retrieved results of a search performed in response to a query has long been proposed as means for improving information delivery [28], especially via the design of cluster-based results interfaces [11, 23, 22, 30]. Ranking (hard) clusters in such interactive retrieval systems, was done, for example, based on the highest query-similarity exhibited by any of the clusters' constituent documents [22]. However, this method, which is well-defined only for hard clustering, always ranks first the single cluster that contains the document most similar to the query [30]. In contrast, our QF property leverages information about the query-similarity of all the cluster's constituent documents; furthermore, our framework is not committed to any specific type of clustering.

Most previous work on ranking (various types of) clusters has focused on comparing a cluster representation with that of the query [12, 4, 34, 16, 24, 26]. We demonstrate the merits of our AllProp algorithm with respect to this approach in Section 4.3.

Some work on re-ranking an initially retrieved list utilizes language models of query-specific clusters to *smooth* document language models [24, 14] so as to improve the document-query similarity estimate. In a related vein, graph-based approaches for re-ranking [7, 17, 18] utilize inter-document similarities information. These approaches can potentially be used to improve our QF estimate, which is based on the document-query "match".

A recently suggested decision procedure for employing either cluster-based or document-based retrieval in response to a query [25] is based on the following observation: clusters containing a high-relevant document percentage exhibit (high) query-similarity that does not deviate much from that of the cluster's constituent documents [25]. Cluster *centrality* [18, 15], as defined by textual similarity to (many) other (central) clusters or documents, its constituent documents' centrality [15], and QF [15], were also shown to be potentially connected to the percentage of relevant documents in a cluster. We compare our AllProp algorithm with some of these approaches for ranking clusters in Section 4.3. However, integrating these cluster properties with the ones we proposed here remains a challenge for future work.

4. EVALUATION

4.1 Language model induction

Unigram language models are the means for producing document rankings in our framework. We now present our methods for inducing a query, a document, and a cluster (unigram) language models — $p_q(\cdot)$, $p_d(\cdot)$, and $p_c(\cdot)$, respectively.

Let $p_x^{MLE}(w)$ be the maximum likelihood estimate (MLE) of term w with respect to the text (or text collection) x. The original query q is used in Eq. 3 to rank documents in the initial list. We represent q by $p_q^{MLE}(\cdot)$. Documents, on the other hand, are text objects that are being ranked (and not "ranked with") in our framework. Therefore, we use $p_d^{Dir[\mu]}(\cdot)$, the Dirichlet-smoothed unigram language model (with smoothing parameter μ) induced from document d [40] for d's representation.

To represent cluster c, we adopt an approach previously proposed for pseudo-feedback-based query expansion [39]. Specifically, we assume that each of c's constituent documents $d \in c$ is generated by a mixture of the cluster topic model $p_c(\cdot)$ — which we want to estimate — with the corpus model. Then, the log-likelihood of the documents in c is

$$\sum_{d_i \in c} \sum_{w} \mathrm{tf}(w \in d_i) \log \left((1-\lambda) p_c(w) + \lambda p_{\mathcal{D}}^{MLE}(w) \right), \quad (8)$$

where $\mathrm{tf}(w \in d_i)$ denotes the number of occurrences of w in d_i [39].

We find $p_c(\cdot)$ by setting λ to a constant and then applying the EM algorithm [39]. In addition, following common practice in work on query expansion in the language model framework [1, 8], we *clip* the cluster language model by setting $p_c(\cdot)$ to zero for all but α terms with the highest $p_c(\cdot)$. (Normalization is then performed to yield a valid probability distribution.) We use $p_c^{Mix[\lambda;\alpha]}(\cdot)$ to denote the resultant cluster language model. Observe that $p_c^{Mix[0;ALL]}(\cdot)$ (where "ALL" means no term clipping) is the MLE of the big document that results from concatenating c's constituent documents; this big document served as a basis for cluster representation in some past work [16, 24, 18].

4.2 Experimental setup

We conducted our experiments on three TREC datasets:

corpus	# of docs	queries	disk(s)
WSJ	173,252	151-200	1-2
TREC8	528,155	401-450	4-5
AP	242,918	51-64, 66-150	1-3

These data sets were also used in some previous work on detecting optimal clusters with which we compare our approach [18]. We applied basic tokenization and Porter stemming via the Lemur toolkit (www.lemurproject.org), which we also used for producing document rankings. We used the titles of TREC topics as queries.

We define $\mathcal{D}_{\text{init}}$, the initially retrieved list upon which clustering is performed, to be $\{\mathcal{L}_{p_q^{MLE}(\cdot)||\mathcal{D}}(i)\}_{i=1}^{50}$ — henceforth referred to as the *initial ranking*. Thus, $\mathcal{D}_{\text{init}}$ is composed of the 50 highest-ranked documents by a standard language-model approach[3].

To produce the set of clusters $Cl(\mathcal{D}_{\text{init}})$, we use a simple nearest-neighbor approach that is known to yield (some) clusters containing a high percentage of relevant documents [14, 18, 25][4]. In fact, the optimal clusters for which the relevant-document percentage is reported in Table 1 (Section 1) are such nearest-neighbor clusters. Specifically, for each $d \in \mathcal{D}_{\text{init}}$, we define a cluster that contains d and its $k-1$ nearest-neighbors in $\mathcal{D}_{\text{init}}$ as measured by $KL\left(p_d^{MLE}(\cdot) \,\middle||\, p_{d_i}^{Dir[\mu]}(\cdot)\right)$ [18]. (Ties are broken by document IDs.) In all the experiments to follow we set k to either 5 or 10.

4.2.1 Evaluation metric, parameters, and efficiency issues

Given a cluster-ranking method, we measure for each query the percentage of relevant documents in the cluster most highly ranked by the method. We use "p@k" to denote this percentage for cluster of size k, because it is exactly the precision at top k documents that is obtained if the constituent documents of the cluster are positioned at the top of the returned document list. (All the presented percentages are averages over a given set of queries.) We use the Wilcoxon two sided test at a confidence level of 95% to determine statistically significant differences of p@k performance.

Our goal in the evaluation to follow is to focus on the underlying principles of our methods. Therefore, rather than engage in excessive parameter tuning, we fix the following parameter values while realizing that the performance attained by our methods is *not necessarily the optimal one they can achieve*: (i) ν, the document-cutoff used by our property-quantification approach (see Section 2) is set to 5000, (ii) μ, the document language model smoothing pa-

[3]To produce an initial ranking of a "reasonable" quality, we set the value of the document-language-model smoothing parameter μ so as to optimize MAP@1000. Such practice also facilitates the comparison to some previous work on optimal-cluster detection that uses the same approach to create an initial list of 50 documents [18].

[4]Although the clusters are overlapping, we only care about the cluster containing the highest relevant-document percentage as we discuss later.

rameter, is set to 2000 following some previous recommendations [40][5], and (iii) α, the number of terms used for cluster representation (see Section 4.1), is set to 50. The single free parameter tuned for our methods is λ, the smoothing parameter that controls cluster representation (see Eq. 8, Section 4.1); we set λ to a value in $\{0, 0.1, \ldots, 0.9\}$ so as to optimize p@k for clusters of size k.

A note on efficiency. We pose our cluster-ranking algorithms as means for obtaining high precision at the very top ranks of the retrieved document list. The users of a system employing our methods should not be concerned with the fact that clustering, or any other computation enhancing the information delivery for that matter, is utilized, as long as the incurred computational overhead is not significant. This is, in fact, the case with our methods. After creating the initial list $\mathcal{D}_{\text{init}}$ by standard retrieval, its clustering can be performed very quickly [38]. (Note that our framework is not committed to any specific clustering algorithm; also, recall that the initial list contains only 50 documents.) Furthermore, creating a cluster model takes only a few iterations of the EM algorithm [39]. Finally, using the (no more than 50) cluster models for quantifying the properties can be implemented as a single run of a single expanded query that contains the terms used for representing all the clusters. (The total number of unique terms used in the clusters' models is relatively small since the clusters are query-specific and we are using only $\alpha = 50$ terms to represent each cluster.) Similar efficiency considerations for using several (query) models were described in some recent reports on predicting query performance and on robust query expansion [36, 3].

4.3 Experimental results

We now present the results of our empirical evaluation. First, we compare our AllProp algorithm with a standard document-based retrieval approach, as well as with previously-proposed (state-of-the-art) approaches for detecting optimal clusters. Then, we present a comparison of AllProp with a state-of-the-art pseudo-feedback-based retrieval method. Finally, we perform some in-depth analysis of the marginal contribution of our four cluster properties and the rank aggregation method that we employ to the overall effectiveness of AllProp.

4.3.1 Main results

Comparison to standard document-based retrieval. In Table 2 we compare the performance of our AllProp algorithm with that of the initial ranking from which $\mathcal{D}_{\text{init}}$ was derived. Clearly, the performance attained by AllProp is substantially better than that of the initial ranking. These results attest to the effectiveness of AllProp in detecting clusters of documents from $\mathcal{D}_{\text{init}}$ that contain a high percentage of relevant documents.

Furthermore, AllProp is superior to document-based retrieval performed over the entire corpus using $KL\left(p_q^{MLE}(\cdot) \,\middle|\middle|\, p_d^{Dir[\mu]}(\cdot)\right)$ with μ optimized for precision-at-top-ranks performance. Case in point, the p@5 performance numbers of such a p@5-optimized baseline are 56.0, 51.2, and 46.5 for WSJ, TREC8 and AP, respectively; the p@10 performance numbers of a p@10-optimized baseline are 49.4, 46.4, and 43.9, respectively.

	WSJ		TREC8		AP	
	p@5	p@10	p@5	p@10	p@5	p@10
init. rank.	53.6	48.4	50.0	45.6	45.7	43.2
AllProp	**62.8***	**53.8**	**54.8**	**49.6**	**49.5**	**45.8**

Table 2: Performance comparison between the AllProp algorithm and the initial ranking. The best result in each column is boldfaced; '*' marks statistically significant difference with the initial ranking.

Comparison to previous methods for optimal-cluster detection. A common approach to ranking (various types of) clusters in response to a query is to measure the similarity of a cluster (as a whole unit) to the query [12, 4, 34, 16, 24, 26]. (We denote this approach by "CQS".) Specifically, CQS was used in the language model framework to rank (hard) query-specific clusters [24]. To compare CQS with our AllProp algorithm, we follow [24] and treat a cluster c as the "big document" that results from concatenating c's constituent documents[6]; we then rank clusters by $KL\left(p_q^{MLE}(\cdot) \,\middle|\middle|\, p_c^{Dir[\mu]}(\cdot)\right)$. ($\mu = 2000$ as at the above.)

In addition, we compare the AllProp algorithm with a recently proposed approach for detecting optimal clusters [18]. The approach is based on defining a one-way bipartite digraph wherein the initial-list ($\mathcal{D}_{\text{init}}$) documents and the query-specific clusters $Cl(\mathcal{D}_{\text{init}})$ form the two sides of the graph. The edges are from each document d to each of its δ nearest-neighbor clusters; the nearest-neighbors of d are determined by $\exp\left(-KL\left(p_d^{MLE}(\cdot) \,\middle|\middle|\, p_{c_i}^{Dir[\mu]}(\cdot)\right)\right)$, which also serves as a weight function for the edges [18]. (Cluster c is represented as at the above, by the "big document" that results from concatenating its constituent documents.) Then, Kleinberg's HITS algorithm [13] is run over the graph, and the clusters are ordered by their induced *authority* scores; indeed, clusters with high authority scores are shown to contain a high percentage of relevant documents [18]. Following [18], we set δ to values in $\{2, 4, 9, 19, 29, 39, 49\}$ so as to optimize p@k performance for clusters of size k; $\mu = 2000$.

Since the reference comparisons just described utilize a concatenation-based representation for clusters (henceforth denoted with "C"), we also test a version of the AllProp algorithm that uses the same cluster representation approach. (We use $p_c^{MLE}(\cdot)$ in Eq. 4-6.) The performance comparison of the methods is presented in Table 3. (We also present for reference the performance of AllProp with the original mixture-model-based representation, denoted with "M".)

Table 3 shows that *both* implementations of AllProp post performance that is better to a statistically significant degree in all relevant comparisons than that of the CQS approach. (Since CQS ranks query-specific clusters based *only* on cluster-query similarity, the performance is not very effective [24, 18].) We can also see that *both* implementations of AllProp outperform the HITS method in most relevant comparisons. These results attest to the effectiveness of the

[5]The only exception is when evaluating the QF property, where we use the same value of μ that was used for creating the initial list $\mathcal{D}_{\text{init}}$.

[6]The order of concatenation has no effect since we only define unigram language models.

	WSJ		TREC8		AP	
	p@5	p@10	p@5	p@10	p@5	p@10
(C) CQS	44.0	37.0	39.6	40.6	39.2	38.8
(C) HITS	53.6^{c}	49.0^{c}	50.8^{c}	46.6	**49.5^{c}**	47.2^{c}
(C) AllProp	61.2^{c}	51.8^{c}	52.0^{c}	48.4^{c}	45.9^{c}	**47.4^{c}**
(M) AllProp	**62.8^{ch}**	**53.8^{ch}**	**54.8^{c}**	**49.6^{c}**	**49.5^{c}**	45.8^{c}

Table 3: Performance comparison of methods for detecting optimal clusters; 'C' and 'M' respectively indicate whether concatenation-based or (our originally proposed) mixture-model-based representation was used for clusters. The best result in each column is boldfaced; statistically significant differences with the CQS [24] and HITS [18] methods are marked with 'c' and 'h', respectively.

AllProp algorithm as a cluster-ranking approach that can effectively accommodate different cluster representations.

Comparison to pseudo-feedback retrieval. To further explore the performance of our AllProp algorithm, we now compare it with that of a state-of-the-art pseudo-feedback-based approach, namely, the *relevance model* [20, 21].

The relevance model RM1 is based on a mixture of the language models of documents in $\mathcal{D}_{\text{init}}$ [21]. Specifically, if w is a term in the vocabulary, $\{q_i\}$ is the set of query terms, and $p_d^{JM[\beta]}(\cdot)$ is a Jelinek-Mercer smoothed document language model with smoothing parameter β [40], then RM1 is defined by

$$p_{RM1}(w;\beta) \stackrel{def}{=} \sum_{d\in\mathcal{D}_{\text{init}}} p_d^{JM[\beta]}(w)\frac{\prod_i p_d^{JM[\beta]}(q_i)}{\sum_{d_j\in\mathcal{D}_{\text{init}}}\prod_i p_{d_j}^{JM[\beta]}(q_i)}.$$

RM1 is often *clipped* by setting $p_{RM1}(w;\beta)$ to 0 for all but the α terms with the highest $p_{RM1}(w;\beta)$ to begin with; further normalization is then performed to yield a valid probability distribution — denoted $\tilde{p}_{RM1}(\cdot;\beta,\alpha)$ [1, 8]. For additional performance enhancement, RM1 is anchored to the original query [1, 8] to yield the RM3 model:

$$p_{RM3}(w;\beta,\alpha,\gamma) \stackrel{def}{=} \gamma p_q^{MLE}(w) + (1-\gamma)\tilde{p}_{RM1}(w;\beta,\alpha);$$

γ is a free interpolation parameter. The documents in the corpus are then ranked by $KL\left(p_{RM3}(\cdot;\beta,\alpha,\gamma)\,\middle\|\, p_d^{Dir[\mu]}(\cdot)\right)$ — the KL divergence of their language models from RM3. (μ is set to 2000 as at the above.)

We (independently) optimize the p@5 and p@10 performance of RM3 by choosing the values of its free parameters from the following sets: (i) $\beta \in \{0, 0.1, 0.3, \ldots, 0.9\}$, (ii) $\alpha \in \{25, 50, 75, 100, 500, 1000, 5000, ALL\}$ where "ALL" stands for all terms in the corpus (i.e., no clipping), and (iii) $\gamma \in \{0, 0.1, 0.2, \ldots, 0.9\}$. The resultant performance is presented in Table 4.

Table 4 shows that our AllProp algorithm outperforms RM3 on WSJ and TREC8, and underperforms it on AP. (None of the performance differences are statistically significant.)[7] However, RM3 posts more statistical significant improvements over the initial ranking than AllProp. All in all, these results are gratifying, because the performance of AllProp was optimized with respect to a single free parameter (λ) , while that of RM3 was optimized with respect to three (β, α and γ).

[7]Similar performance patterns are observed if we use RM3 for re-ranking $\mathcal{D}_{\text{init}}$, rather than for ranking the entire corpus. Specifically, the optimized p@5 of such a model is 58.8, 53.6, and 51.1 for WSJ, TREC8, and AP, respectively; the optimized p@10 is 53.0, 48.4, and 48.3, respectively.

	WSJ		TREC8		AP	
	p@5	p@10	p@5	p@10	p@5	p@10
init. rank.	53.6	48.4	50.0	45.6	45.7	43.2
RM3	58.8*	53.4*	53.6	48.2	**50.3***	**48.6***
AllProp	**62.8***	**53.8**	**54.8**	**49.6**	49.5	45.8

Table 4: Comparison of our AllProp algorithm with a relevance model (RM3) [1, 8]. Boldface: best performance in a column; '*': statistically significant difference with the initial ranking.

Perhaps not less important is the following observation. Recall that AllProp looks for relevant documents (via cluster ranking) in an initially retrieved list $\mathcal{D}_{\text{init}}$, while pseudo-feedback-based methods such as RM3 use documents from $\mathcal{D}_{\text{init}}$ for defining a (query) model using which the entire corpus is (re-)ranked. Thus, in principle, AllProp can be employed as a "filter" on $\mathcal{D}_{\text{init}}$, providing pseudo-feedback-based approaches with a "relevance-focused" subset of $\mathcal{D}_{\text{init}}$ upon which they can operate. In fact, the potential merits of such integration were demonstrated in some previous work on implementing relevance-feedback on top of documents selected from *manually chosen* query-specific clusters [30]. Exploring this direction is thus a promising venue for future research on pseudo-feedback retrieval.

4.3.2 Further analysis

Heretofore we have focused on the AllProp algorithm, which ranks clusters based on the extent to which they exhibit all four "faithfulness" properties that were defined in Section 2. Having read this far, the reader may rightfully wonder whether all these properties in fact contribute to the overall performance of the AllProp algorithm. Here we show that the answer to this question is affirmative, and even to a quite surprising extent.

Table 5 shows the resultant performance of using each of our four properties individually, as well as using all their possible combinations (based on Eq. 7). When using each of the properties by itself, PF turns out to be the most effective such property. In fact, it is the only single property that consistently yields performance superior to that of the initial ranking (from which $\mathcal{D}_{\text{init}}$ was derived) in all relevant comparisons. This finding attests to the importance of measuring the (targeted by PF) extent to which a cluster exhibits different "aspects" in the initial list, as manifested in its clustering.

Note that the other three properties, namely, QF, SF, and ILF, appear to be rather noisy when used in isolation (as their performance is often below that of the initial ranking), and thus these results as if suggest that using these properties is likely to harm the overall ability to detect the desired clusters. Table 5, however, shows that this is very much *not* the case. Indeed, perhaps the most interesting observation that can be made based on Table 5 is that in all six relevant comparisons (3 corpora × 2 evaluation measures) the performance mostly improves from using any subset of properties X to any of its proper extensions $X \cup \{\mathcal{P}\}$, where $\mathcal{P}$ is a single property in $\{QF, SF, ILF, PF\}$. More specifically,

	WSJ		TREC8		AP	
	p@5	p@10	p@5	p@10	p@5	p@10
init. rank.	53.6	48.4	50.0	45.6	45.7	43.2
QF	55.2	49.2	46.8	49.6	42.6	47.0*
ILF	46.0	41.4*	40.0*	36.4*	39.6*	37.0*
SF	52.0	50.4	44.4	48.0	43.2	45.1
PF	56.0	51.2	50.4	46.6	48.7	46.2
QF ∧ SF	59.6	53.6*	50.8	49.8	45.1	45.3
QF ∧ ILF	57.2	51.2	50.8	48.4	44.0	44.6
QF ∧ PF	59.2	51.4	53.2	48.8	47.5	**47.4**
SF ∧ ILF	52.4	52.8	48.0	48.0	47.5	43.6
SF ∧ PF	55.6	51.2	52.4	49.2	45.9	45.4
ILF ∧ PF	56.4	50.8	55.2	48.8	47.1	45.9
QF ∧ SF ∧ ILF	59.2	53.2*	54.0	**51.8***	47.3	46.1
QF ∧ SF ∧ PF	**62.8***	52.6	53.6	49.4	48.1	**47.4**
QF ∧ ILF ∧ PF	61.2*	51.8	55.6	48.8	48.3	47.0
SF ∧ ILF ∧ PF	58.4	51.6	**56.8**	49.6	45.9	45.5
AllProp	**62.8***	**53.8**	54.8	49.6	**49.5**	45.8

Table 5: The performance of ranking clusters by single cluster-properties, as well as by their combinations (using Eq. 7). Recall that AllProp stands for QF ∧ SF ∧ ILF ∧ PF. Best result in each column is boldfaced; statistically significant differences with the initial ranking are marked with '*'.

- The performance either remains the same or improves in 83% of such extensions of X to $X \cup \{\mathcal{P}\}$. Moreover, across the relevant comparisons, the absolute best-case improvements in p@k are more substantial (between +9% and +18%) than the worst-case such degradations in p@k (between -0.4% and -2.8%).
- Even the worst-performing single property, ILF,[8] helps to improve the performance in about 80% of the extensions of X to $X \cup \{ILF\}$ as above. In short, "very noisy" does not mean "uninformative"!
- Considering the few cases wherein the extension of X to $X \cup \{\mathcal{P}\}$ results in performance degradation, no single property of the clusters suggests itself as a clear "destroyer". Specifically, the properties QF, SF, ILF, and PF partition these cases by their responsibility for being the degrading $\mathcal{P}$ into 11%, 32%, 28.5%, and 28.5%, respectively.
- For all pairs of properties $\mathcal{P}, \mathcal{P}'$ in $\{QF, SF, ILF, PF\}$, there appears to be no visible symmetry in the performance change between extending a subset of properties X to $X \cup \{\mathcal{P}\}$ and to $X \cup \{\mathcal{P}'\}$. This suggests that the cluster-properties we have defined are effectively treated by the rank aggregation process as "pseudo-independent" classifiers of the query-specific clusters.
- Last but not least, our AllProp algorithm that integrates all four properties is superior in most relevant comparisons to using any subset of the four.

In light of the demonstrated effectiveness of aggregating estimates of the different witnessing properties of the clusters, we now turn to study some alternative rank-aggregation approaches. Recall from Section 2 that our original rank-aggregation method for property integration uses the algebraic product of the per-property scores of clusters. (See Eq. 7.) We first explore an intuitive alternative that uses sum [10] instead of product:

$$Score_{\vee}(c;\Pi) \stackrel{def}{=} \sum_{\mathcal{P}\in\Pi} Score_{\mathcal{P}}(c),$$

where $\Pi = \{QF, SF, ILF, PF\}$. This aggregation procedure has a more "disjunctive" nature — $Score_{\vee}(\cdot;\cdot)$ and $Score_{\wedge}(\cdot;\cdot)$ are ranking-wise equivalents of the arithmetic and geometric means of the individual scores, respectively.

Now, suppose that we are to believe that the only semantics of our ranking functions is ordinal, that is, either the absolute scores have no meaning, or this meaning significantly varies among the four ranking functions. If so, then sometimes the absolute scores are better ignored, and we should resort to one of the ordinal rank aggregation methods [6, 10]. For instance, one of the most popular such methods is Borda's method [37], which in our context, scores cluster c by:

$$Score_{Borda}(c;\Pi) \stackrel{def}{=} \sum_{\mathcal{P}\in\Pi} \sum_{c'\in Cl(\mathcal{D}_{\text{init}})} \delta\big[Score_{\mathcal{P}}(c) > Score_{\mathcal{P}}(c')\big],$$

where $\Pi = \{QF, SF, ILF, PF\}$; then, the clusters are ranked in descending order of their Borda scores. Similarly to product-based or sum-based score aggregations, "positional" methods such as Borda's are appealing because they are computationally easy[9]. In fact, note that Borda's method boils down to sum-based score aggregation of certain, inter-comparable between the individual properties, yet basically ad hoc scores of the clusters. Hence, if the scores provided by different cluster-ranking properties have similar quantitative nature — as is the case with our average-precision-based measures of the four "faithfulness" criteria — then *a priori* Borda's method is less appealing.

	WSJ		TREC8		AP	
	p@5	p@10	p@5	p@10	p@5	p@10
init. rank.	53.6	48.4	50.0	45.6	45.7	43.2
∧	**62.8***	53.8	**54.8**	49.6	**49.5**	45.8
∨	59.2	52.0	50.4	**51.0***	43.2	**46.7**
Borda	59.6	**54.4***	54.4	49.8	48.9	46.2

Table 6: Comparison of rank-aggregation methods: product (∧, as in our AllProp algorithm), sum (∨), and Borda's method. Boldface marks the best performance in each column, and '*' marks statistically significant difference with the initial ranking.

Table 6 provides a performance comparison between the aforementioned three rank-aggregation methods. Carefully looking into the numbers, it appears that both the product-based method (∧) employed by our AllProp algorithm and Borda's method are superior to the sum-based method (∨) in most of the relevant comparisons. (While ∧ is superior to Borda for clusters of size 5, the reverse is true for clusters of size 10.) Having said that, we believe that the more

[8]The fact that the list-faithfulness property, ILF, is the worst performing (used alone) property can be attributed to the fact that for many queries, many of the documents in $\mathcal{D}_{\text{init}}$ are not relevant; hence, imposing faithfulness to these documents (without using other properties) yields degraded performance.

[9]Depriving from positional methods very quickly brings us to NP-hard methods such as Kemeny optimal aggregation, footfule optimal aggregation, etc. [9].

important observation is that all three rank-aggregation approaches yield performance that transcends (in most relevant comparisons) that of the initial ranking that was used to create $\mathcal{D}_{\text{init}}$. Thus, it seems that using cluster-properties of a somewhat "complementary" nature, as is the case for our properties (see at the above), yields satisfactory performance with different rank-aggregation approaches.

5. SUMMARY AND FUTURE WORK

We presented a novel framework to ranking query-specific clusters by the presumed percentage of relevant documents that they contain. The framework is based on proposing cluster properties that presumably correlate with this percentage, and integrating their estimates via rank aggregation. We showed that our approach outperforms previous methods for detecting clusters containing a high relevant-document percentage, and also produces document-ranking with substantially higher precision-at-top-ranks performance than that of standard document-based retrieval.

We intend to integrate additional cluster properties in our framework [25, 18, 15] and to study alternative rank-aggregation methods. Another future direction is to introduce *diversity* in the resultant list of documents, which is important for handling ambiguous queries. Using documents from different presumably-optimal clusters can potentially be effective in such cases.

Acknowledgments We thank the reviewers for their comments. This paper is based upon work supported in part by a gift from Google. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsoring institutions.

6. REFERENCES

[1] N. Abdul-Jaleel, J. Allan, W. B. Croft, F. Diaz, L. Larkey, X. Li, M. D. Smucker, and C. Wade. UMASS at TREC 2004 — novelty and hard. In *Proceedings of TREC-13*, 2004.

[2] C. Buckley, G. Salton, J. Allan, and A. Singhal. Automatic query expansion using SMART: TREC3. In *Proceedings of TREC-3*, pages 69–80, 1994.

[3] K. Collins-Thompson and J. Callan. Estimation and use of uncertainty in pseudo-relevance feedback. In *Proceedings of SIGIR*, pages 303–310, 2007.

[4] W. B. Croft. A model of cluster searching based on classification. *Information Systems*, 5:189–195, 1980.

[5] W. B. Croft and J. Lafferty, editors. *Language Modeling for Information Retrieval*. Number 13 in Information Retrieval Book Series. Kluwer, 2003.

[6] P. Diaconis. *Group Theory in Statistics*. Harvard Lecture Notes, 1982.

[7] F. Diaz. Regularizing ad hoc retrieval scores. In *Proceedings of CIKM*, pages 672–679, 2005.

[8] F. Diaz and D. Metzler. Improving the estimation of relevance models using large external corpora. In *Proceedings of SIGIR*, pages 154–161, 2006.

[9] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the Web. In *Proceedings of the World Wide Web Conference*, pages 613–622, Hong Kong, 2001.

[10] E. A. Fox and J. A. Shaw. Combination of multiple searches. In *Proceedings of TREC-2*, 1994.

[11] M. A. Hearst and J. O. Pedersen. Reexamining the cluster hypothesis: Scatter/Gather on retrieval results. In *Proceedings of SIGIR*, pages 76–84, 1996.

[12] N. Jardine and C. J. van Rijsbergen. The use of hierarchic clustering in information retrieval. *Information Storage and Retrieval*, 7(5):217–240, 1971.

[13] J. Kleinberg. Authoritative sources in a hyperlinked environment. Technical Report Research Report RJ 10076, IBM, May 1997.

[14] O. Kurland. *Inter-document similarities, language models, and ad hoc retrieval*. PhD thesis, Cornell University, 2006.

[15] O. Kurland. The opposite of smoothing: A language model approach to ranking query-specific document clusters. In *Proceedings of SIGIR*, 2008.

[16] O. Kurland and L. Lee. Corpus structure, language models, and ad hoc information retrieval. In *Proceedings of SIGIR*, pages 194–201, 2004.

[17] O. Kurland and L. Lee. PageRank without hyperlinks: Structural re-ranking using links induced by language models. In *Proceedings of SIGIR*, pages 306–313, 2005.

[18] O. Kurland and L. Lee. Respect my authority! HITS without hyperlinks utilizing cluster-based language models. In *Proceedings of SIGIR*, pages 83–90, 2006.

[19] J. D. Lafferty and C. Zhai. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of SIGIR*, pages 111–119, 2001.

[20] V. Lavrenko and W. B. Croft. Relevance-based language models. In *Proceedings of SIGIR*, pages 120–127, 2001.

[21] V. Lavrenko and W. B. Croft. Relevance models in information retrieval. In Croft and Lafferty [5], pages 11–56.

[22] A. Leuski. Evaluating document clustering for interactive information retrieval. In *Proceedings of CIKM*, pages 33–40, 2001.

[23] A. Leuski and J. Allan. Evaluating a visual navigation system for a digital library. In *Proceedings of ECDL*, pages 535–554, 1998.

[24] X. Liu and W. B. Croft. Cluster-based retrieval using language models. In *Proceedings of SIGIR*, pages 186–193, 2004.

[25] X. Liu and W. B. Croft. Experiments on retrieval of optimal clusters. Technical Report IR-478, Center for Intelligent Information Retrieval (CIIR), University of Massachusetts, 2006.

[26] X. Liu and W. B. Croft. Representing clusters for retrieval. In *Proceedings of SIGIR*, pages 671–672, 2006. Poster.

[27] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *Proceedings of SIGIR*, pages 275–281, 1998.

[28] S. E. Preece. Clustering as an output option. In *Proceedings of the American Society for Information Science*, pages 189–190, 1973.

[29] I. Ruthven and M. Lalmas. A survey on the use of relevance feedback for information access systems. *Knowledge Engineering Review*, 18(2):95–145, 2003.

[30] J. G. Shanahan, J. Bennett, D. A. Evans, D. A. Hull, and J. Montgomery. Clairvoyance Corporation experiments in the TREC 2003. High accuracy retrieval from documents (HARD) track. In *Proceedings of TREC-12*, pages 152–160, 2003.

[31] A. Tombros, R. Villa, and C. van Rijsbergen. The effectiveness of query-specific hierarchic clustering in information retrieval. *Information Processing and Management*, 38(4):559–582, 2002.

[32] C. J. van Rijsbergen. *Information Retrieval*. Butterworths, second edition, 1979.

[33] E. M. Voorhees and D. K. Harman. *TREC: Experiments and evlautaion in information retrieval*. The MIT Press, 2005.

[34] P. Willett. Query specific automatic document classification. *International Forum on Information and Documentation*, 10(2):28–32, 1985.

[35] J. Xu and W. B. Croft. Query expansion using local and global document analysis. In *Proceedings of SIGIR*, pages 4–11, 1996.

[36] E. Yom-Tov, S. Fine, D. Carmel, and A. Darlow. Learning to estimate query difficulty: including applications to missing content detection and distributed information retrieval. In *Proceedings of SIGIR*, pages 512–519, 2005.

[37] H. P. Young. An axiomatization of Borda's rule. *Journal of Economic Theory*, 9:43–52, 1974.

[38] O. Zamir and O. Etzioni. Web document clustering: a feasibility demonstration. In *Proceedings of SIGIR*, pages 46–54, 1998.

[39] C. Zhai and J. D. Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of CIKM*, pages 403–410, 2001.

[40] C. Zhai and J. D. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of SIGIR*, pages 334–342, 2001.

[41] H. J. Zimmermann. *Fuzzy Set Theory*. Kluwer Academic, 3 edition, 1996.