Integrating Clusters Created Offline with Query-Specific Clusters for Document Retrieval

Lior Meister, Oren Kurland, and Inna Gelfer Kalmanovich
Faculty of Industrial Engineering and Management
Technion — Israel Institute of Technology
Haifa 32000, Israel
meister@tx.technion.ac.il,kurland@ie.technion.ac.il,innagel@tx.technion.ac.il

ABSTRACT

Previous work on cluster-based document retrieval has used *either* static document clusters that are created offline, or query-specific (dynamic) document clusters that are created from top-retrieved documents. We present the potential merit of integrating these two types of clusters.

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

General Terms: Algorithms, Experimentation

Keywords: cluster-based retrieval, re-ranking

1. INTRODUCTION

Utilizing information induced from clusters of similar documents has long been suggested as a means to improving document-retrieval effectiveness [1].

Cluster-based document-retrieval models utilize mainly two types of clustering. The first is *static* (offline) clustering performed prior to retrieval time [1, 3, 4]. Information induced from static clusters is used to rank the corpus in response to a query. Since the clustering is query-independent, the clusters that presumably pertain to a given query have to be identified at retrieval time. A potential merit of using static clustering, for example, is that relevant documents that exhibit low surface-level query-similarity can still be identified if they are associated with (belong to) clusters with documents that exhibit high query-similarity. Thus, *vocabulary mismatch* between a query and relevant documents can be addressed.

Dynamic (query-specific) clusters are created from the documents most highly ranked by an initial search performed in response to a query [6, 4]. The clusters are often used to re-rank these initially highly-ranked documents so as to improve precision at the very top ranks [2]. In contrast to static clusters, dynamic clusters need to be computed for each query. Furthermore, they do not contain documents exhibiting very low surface-level query similarity. Nevertheless, dynamic clusters are more "query-focused" than static clusters by the virtue of the way they are created, and hence, are somewhat less likely to exhibit *query drift* [5].

Static and dynamic clusters were used separately in work on cluster-based document retrieval. We demonstrate the potential merit of integrating the two types of clusters.

SIGIR'09, July 19–23, 2009, Boston, Massachusetts, USA.

2. RETRIEVAL FRAMEWORK

Let q, d, and $\mathcal{D}$ denote a query, a document, and a corpus, respectively.

Dynamic clusters have been used for the re-ranking task [6, 4, 2]. That is, re-ordering the documents in an initial list, $\mathcal{D}_{init}$, that was retrieved in response to q by some search algorithm so as to improve precision at top ranks. Hence, we focus on this task, and present methods that rank only documents d in $\mathcal{D}_{init}$.

Our algorithms operate in the language modeling framework. We use $p_x(y)$ to denote the language-model-based similarity between texts x and y.[1] (A text can be a query, a document, or a cluster of documents; we describe the estimate in Section 3.)

We assume that the corpus $\mathcal{D}$ is clustered prior to retrieval time into *static* clusters. We use S to denote the set of m_s static clusters c that exhibit the highest query-similarity $p_c(q)$. Hence, while static clusters are created in a query-independent fashion, we use those that could be considered as "query-related". In addition, we cluster $\mathcal{D}_{init}$, the list to be re-ranked, into the set T of m_t *dynamic* clusters.

The re-ranking methods that we present are based on the *interpolation* algorithm [3, 2]. The algorithm was shown to yield state-of-the-art performance when using static clusters to rank the entire corpus and when using dynamic clusters to re-rank $\mathcal{D}_{init}$. Specifically, if X is a set of clusters, the retrieval score of document d is: $\lambda p_d(q) + (1 - \lambda) \sum_{c \in X} p_c(q) p_d(c)$; λ is a free parameter. Note that clusters serve as proxies for d for estimating query similarity.

Re-ranking algorithms. A previously-proposed re-ranking algorithm [2], **Interp**($\mathbf{T}$), utilizes only query-specific clusters as proxies for d: $\lambda p_d(q) + (1 - \lambda) \sum_{t \in T} p_t(q) p_d(t)$. Another reference comparison that we consider is the **Interp**($\mathbf{S}$) method [3] that uses only static clusters as proxies for d: $\lambda p_d(q) + (1 - \lambda) \sum_{s \in S} p_s(q) p_d(s)$. Naturally, then, we can use both sets of clusters as proxies for d so as to leverage the merits of each (refer back to Section 1). The resul-

[1] $p_x(y)$ can be thought of as a surrogate for the probability $p(y|x)$ often used in work on language models for information retrieval; refer to [3] for more details.

corpus	queries	disk(s)
ROBUST	301-450, 601-700	4,5 (-CR)
AP	51-150	1-3
WSJ	151-200	1-2
SJMN	51-150	3

	ROBUST		AP		WSJ		SJMN	
	p@5	p@10	p@5	p@10	p@5	p@10	p@5	p@10
init. rank.	48.2	43.2	44.6	43.5	55.6	49.8	33.2	28.7
opt. base.	48.3	44.0^{i}	45.5	43.5	57.6	50.0	33.6	28.7
Interp(T)	50.2^{i}	45.1^{i}	50.1^{i}	47.8^{io}	60.8	53.4^{i}	36.0	32.9^{io}
Interp(S)	50.0	$\mathbf{45.7}^{io}$	51.3^{io}	49.3^{io}	60.8	$\mathbf{57.0}^{io}$	37.4^{i}	33.8^{io}
Interp($T \rightarrow S$)	49.3	45.1^{i}	49.3	48.2^{io}	60.8^{i}	54.6^{io}	37.4	33.7^{io}
Interp($S \rightarrow T$)	50.0	45.0^{i}	51.5^{io}	$\mathbf{49.8}^{io}$	61.6	55.0^{io}	$\mathbf{39.0}^{io}$	$\mathbf{35.0}^{io}_{t}$
Interp($S \cup T$)	$\mathbf{50.9}^{io}$	45.2^{i}	$\mathbf{51.7}^{io}$	49.6^{io}	$\mathbf{62.0}^{i}$	55.4^{io}	38.4^{i}	34.6^{io}_{t}

Table 1: Corpora used for experiments and performance numbers. The best result in a column is boldfaced. Statistically significant differences with the initial ranking (init. rank.) of $\mathcal{D}_{init}$, the optimized baseline (opt. base.), and Interp(T), are marked with 'i', 'o', and 't', respectively. (There are no statistically-significant differences between the performance of Interp(S) and that of the models that utilize both S and T.)

tant algorithm, **Interp($\mathbf{S \cup T}$)**, scores d by: $\lambda p_d(q) + (1 - \lambda) \sum_{c \in S \cup T} p_c(q) p_d(c)$.

Another approach that we propose here is based on utilizing inter-cluster similarities. Specifically, we let one type of clusters serve as a proxy for the other type for estimating cluster-query and document-cluster similarities. For example, the **Interp($\mathbf{T \rightarrow S}$)** algorithm uses dynamic clusters as proxies for d and static clusters as proxies for dynamic clusters: $\lambda p_d(q) + (1 - \lambda) \sum_{s \in S, t \in T} p_s(q) p_t(s) p_d(t)$. Analogously, **Interp($\mathbf{S \rightarrow T}$)** uses static clusters as proxies for d and dynamic clusters as proxies for static clusters: $\lambda p_d(q) + (1 - \lambda) \sum_{s \in S, t \in T} p_t(q) p_s(t) p_d(s)$.

3. EVALUATION

The TREC corpora specified in Table 1 were used for experiments. We applied tokenization, Porter stemming, and stopword removal (using the INQUERY list) via the Lemur toolkit (www.lemurproject.org), which was also used for language-model induction. Topic titles served as queries.

We use a previously-proposed language-model-based similarity estimate, which was shown to be effective for cluster-based retrieval [3]. Specifically, let $p_z^{Dir[\mu]}(\cdot)$ be the Dirichlet-smoothed unigram language model (with smoothing parameter μ) induced from text z. Then, $p_x(y) \stackrel{def}{=} \exp\left(-D\left(p_y^{Dir[0]}(\cdot) \,\middle\|\, p_x^{Dir[\mu]}(\cdot)\right)\right)$, where D is the KL divergence. Unless otherwise specified, we set $\mu = 1000$.

Since the goal of re-ranking methods is to improve precision at top ranks, we use the precision of the top 5 and 10 documents (p@5, p@10) for evaluation measures. Statistically-significant differences of performance are determined using the Wilcoxon two-tailed test at a 95% confidence level.

Following previous work on re-ranking [2], we set the list $\mathcal{D}_{init}$ upon which re-ranking is performed to the 50 documents d in the corpus that yield the highest $p_d(q)$ (i.e., standard language-model-based retrieval), where μ is set to optimize MAP; hence, the **initial ranking** of $\mathcal{D}_{init}$ is of high quality. As a reference comparison we use an **optimized baseline** that ranks the corpus by $p_d(q)$ with μ set to optimize p@5. The value of λ is chosen for each re-ranking method from $\{0, 0.1, \ldots, 0.9\}$ to optimize p@5.

Each of the cluster-sets S and T is composed of $m_s = m_t = 50$ nearest-neighbors-based clusters of 10 documents that are created using the $p_x(y)$ similarity estimate [3, 2]. Studying the effect of the number of clusters, and the cluster size, on performance is left for future work. A cluster is represented by the concatenation of its constituent documents. The order of concatenation has no effect since we use unigram language models that assume term independence.

We can see in Table 1 that both the algorithms that use either static (S) or dynamic (T) clusters, and our newly-proposed algorithms that integrate the two, are highly effective in re-ranking. Indeed, the performance of the methods transcends that of the initial ranking and the optimized baseline — often to a statistically significant degree — in most relevant comparisons (corpus × evaluation measure).

We can also see that our Interp($S \rightarrow T$) and Interp($S \cup T$) algorithms outperform Interp(T) and Interp(S) in most relevant comparisons. (However, the differences are rarely statistically significant.) Furthermore, Interp($S \cup T$) is the only algorithm among those in Table 1 that posts statistically-significant improvements over the initial ranking for all relevant comparisons. These findings attest to the merits of utilizing *both* static and dynamic clusters.

Finally, we postulate that the performance of Interp($T \rightarrow S$) is inferior to that of Interp($S \rightarrow T$) and Interp($S \cup T$) because Interp($T \rightarrow S$) uses static *query-independent* clusters as proxies for dynamic query-specific clusters for estimating cluster-query similarities.

4. CONCLUSION AND FUTURE WORK

We demonstrated the potential merits of integrating static (offline created) and dynamic (query-specific) document clusters for ad hoc document retrieval. For future work we plan on devising additional integration methods.

Acknowledgments We thank the reviewers for their comments, and Lillian Lee for discussions of ideas presented in this paper. The paper is based upon work supported in part by IBM's and Google's faculty research awards. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsors.

5. REFERENCES

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

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

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

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

[5] M. Mitra, A. Singhal, and C. Buckley. Improving automatic query expansion. In *Proceedings of SIGIR*, pages 206–214, 1998.

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