Efficient Summarization-Aware Search for Online News Articles

Wisam Dakka
Computer Science Department
Columbia University
wisam@cs.columbia.edu

Luis Gravano
Computer Science Department
Columbia University
gravano@cs.columbia.edu

ABSTRACT

News portals gather and organize news articles published daily on the Internet. Typically, news articles are clustered into "events" and each cluster is displayed with a short description of its contents. A particularly interesting choice for describing the contents of a cluster is a machine-generated multi-document summary of the articles in the cluster. Such summaries are informative and help news readers to identify and explore only clusters of interest. Naturally, multi-document clusters and summaries are also valuable to help users navigate the results of keyword-search queries. Unfortunately, current document summarizers are still slow; as a result, search strategies that define document clusters and their multi-document summaries online, in a query-specific manner, are prohibitively expensive. In contrast, search strategies that only return offline, query-independent document clusters are efficient, but might return clusters whose (query-independent) summaries are of little relevance to the queries. In this paper, we present an efficient *Hybrid* search strategy to address the limitations of fully online and fully offline summarization-aware search approaches. Extensive experiments involving user relevance judgments and real news articles show that the quality of our *Hybrid* results is high, and that these results are computed in substantially less time than with the fully online strategy. We have implemented our strategy and made it available on the Newsblaster news summarization system, which crawls and summarizes news articles from a variety of web sources on a daily basis.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Clustering, Search process*; H.m [**Miscellaneous**]: Multi-document Summarization

General Terms

Algorithms, Performance, Design, Experimentation

Keywords

Summarization-Aware Search, Summarization, Clustering

JCDL '07, June 17–22, 2007, Vancouver, British Columbia, Canada.

1. INTRODUCTION

News portals gather and organize news articles published daily on the Internet. As a notable example, Google News[1] continuously crawls news sites such as The New York Times to extract stories. Google News then categorizes stories into a few broadly defined areas, groups stories about the same event into clusters, and extracts the lead of one story about each event as the event's "snippet." At the end of this process, news stories from a few thousand diverse sources have been organized into clusters, and each cluster is displayed with an image, a title, and a snippet, derived from the associated articles.

A natural next step to enrich a portal's content is to use machine summarization. As a well known example, the Newsblaster system [16] clusters news articles into events and then produces a short machine-generated summary of the multiple documents in each cluster. Another example of this kind of systems is NewsInEssence [17], which also provides on-the-fly personalized clusters and summaries. A key difference between Google News and systems such as Newsblaster and NewsInEssence is that the latter intensively use natural language processing (NLP) to generate their multi-document summaries, while Google News just extracts a sentence from a story on an event as the event summary. Producing the NLP summaries, which are generally informative and high-quality, is time consuming, and the generation of a multi-document summary often takes on the order of one minute in operational systems. In these systems, most summarization-related computation is performed overnight, with incremental updates throughout the day.

News portals can incorporate "traditional" keyword search capabilities and return a ranked list of documents for a query, as is the case in Yahoo! News. In some other portals, the result for a query consists of *offline*, query-independent document clusters[2] that somehow "match" the query. An alternative that we explore in this paper is to *fully integrate keyword search with multi-document clustering and multi-document summarization.* In such *summarization-aware search*, query results are clusters about events relevant to the query. Similar to Newsblaster and NewsInEssence, these clusters are accompanied with machine-generated summaries. In addition, such results should be computed efficiently and should be at least as accurate, as we will discuss, as in traditional search.

EXAMPLE 1. Figure 1 shows the top-3 clusters in the results for the query *[Hurricane Wilma]* produced using a summarization-aware search system that we will define later. Each cluster includes several stories (from different sources) that are likely to cover the same news event. Each result cluster has an associated summary, capturing the gist of the event stories.□

[1] news.google.com

[2] Google News returns a ranked list of offline clusters after filtering out the non-matching documents from them.

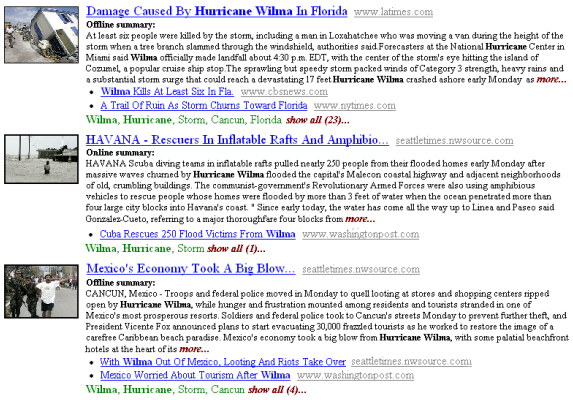


Figure 1: Top-3 clusters in a summarization-aware search result for the query *[Hurricane Wilma]*.

Producing summarization-aware search results for a query over a collection of news articles involves multiple challenges, including: (1) identifying the appropriate article clusters; (2) generating a multi-document summary for each cluster; and (3) ranking the clusters on query relevance. Furthermore, we need to perform all these tasks efficiently.

Two natural techniques for summarization-aware search suggest themselves: either we can return static, offline clusters and their summaries (which are, say, computed overnight) or, alternatively, we can perform online clustering and summarize the articles that match the query at query-execution time. These *fully offline* and *fully online* techniques differ widely in the amount of computation required at query-execution time and in the quality of the returned results. While the first technique is efficient and produces clusters that correspond to real news events, it lacks in producing query-specific clusters with query-specific summaries. The second technique is likely to produce high-quality results, as has been suggested in numerous studies; unfortunately, the online summarization required by this technique results in unacceptably long response times, as we will show in Section 4.3.

To address the limitations of the fully offline and online approaches, in this paper we present a novel *Hybrid* technique that borrows elements from the two search approaches. Our approach is *summarizer-independent*: we interact with the summarization system as a black box and investigate to what extent we can reduce the time for producing summarization-aware search results without modifying the underlying summarizer.

To evaluate the relative merits of (many variations of) the different search techniques, we performed extensive experiments using Newsblaster as the state-of-the-art multi-document summarization system of choice, including user relevance judgments and real news articles. For our experiments, we collected 184 queries issued by 22 users over six days' worth of news articles, with 13,017 news stories from 18 U.S. news sources organized into 1,620 clusters. We compare our *Hybrid* technique against both cluster-based (e.g., the above offline and online approaches) and document-based search strategies. Our experiments show that the *Hybrid* strategy produces results that are comparable in quality to those from state-of-the-art cluster-based techniques. Furthermore, unlike the fully online technique, the *Hybrid* technique has moderate response times. Our experiments also suggest that the results from the *Hybrid* strategy are at least as accurate as those from state-of-the-art document-based techniques.

The focus of this paper is on the efficiency of a particular family of summarization-aware search techniques, where the query results are document clusters with NLP-derived multi-document summaries. We do not consider less costly query result presentation or summarization alternatives outside of this summarization-aware search family (e.g., we do not analyze systems such as Google News where cluster summaries consist of just the document titles or the leading sentence of a "central" document in a cluster). Extending our analysis to other families of query result presentation or summarization approaches is the subject of interesting future work.

The rest of the paper is organized as follows. Section 2 elaborates on the offline and online summarization-aware search approaches described above, which serve as our "baseline" techniques. Section 3 presents our novel *Hybrid* strategy, which requires addressing two main challenges. First, we study the problem of identifying the offline clusters that are relevant to a query. Second, we propose alternatives to rank cluster-based search results. Section 4 reports the experimental settings and results. Finally, Sections 5 and 6 describe related work and conclude the paper.

2. BASELINE SEARCH APPROACHES

We now describe two baseline *summarization-aware* search techniques. Specifically, the first baseline technique only returns offline article clusters and summaries (Section 2.1), while the second technique creates the article clusters and summaries fully online, in a query-specific way (Section 2.2). Later, in Section 3, we will show how we can combine these two techniques and exploit their advantages to define our *Hybrid* search strategy.

2.1 Offline Summarization and Clustering

A natural technique for answering queries in a summarization-aware way is to return merely "old" clusters that are produced offline prior to the actual search. This inexpensive technique processes a query as follows: (1) uses a search criterion on the offline clusters to select an initial set of clusters for the query, (2) applies a cluster-ranking criterion to order the selected clusters, and (3) returns the top-k clusters for the query from the ordered set, for some predetermined value of k (e.g., $k = 3$).

This technique might return clusters on events that are irrelevant to the query, with stories that are relevant to the query but that are not central to the clusters' focus, and hence many other irrelevant stories might be present in the returned clusters as well. If we do not include such clusters in the query results, we ignore relevant articles in them, hurting recall; if, alternatively, we include the clusters in the results, we hurt precision and introduce offline clusters whose query-independent summaries might be irrelevant to the query. This is a fundamental problem with this technique, which motivates the introduction of alternatives.

2.2 Online Summarization and Clustering

Another natural technique for answering a query in a summarization-aware way is to return "new" clusters that are produced on-the-fly after identifying the documents matching the query. Unlike the previous technique, all stories in these clusters match the user query; therefore, precision is anticipated to be high. More specifically, this expensive technique processes a query as follows: (1) searches for documents that match the query, (2) performs clustering on the matching documents to create an initial set of clusters, (3) applies a cluster-ranking criterion to order the new clusters, (4) summarizes the top-k clusters for the query from the ordered set, and (5) returns the newly summarized clusters.

This technique is likely to produce high-quality results. It honors the cluster hypothesis [20], which states that relevant documents

tend to be more similar to each other than to irrelevant ones. Furthermore, Hearst and Pedersen [5], among others, have claimed that the precision of query-specific, cluster-based search results is promising, while Tombros et al. [19] have showed that online clustering of query results significantly outperformed an offline cluster-based retrieval similar to that in Section 2.1. Unfortunately, state-of-the-art multi-document summarizers are slow since they rely heavily on inherently slow NLP tools for parsing, tagging, and discourse and sentence analysis [13]. This compromises the applicability of online approaches in practice for efficiency considerations.

3. HYBRID SEARCH

This paper focuses on providing a summarization-aware search interface over a news portal. We assume that, for browsing, such a portal organizes the articles in query-independent *offline* clusters (e.g., as Google News does), and that each offline cluster has an associated multi-document summary (e.g., as Newsblaster and NewsInEssence provide). In this section, we introduce our *Hybrid* search strategy, which exploits the offline document clusters and summaries, when appropriate, to produce query-specific summarization-aware search results efficiently.

3.1 Overview

Unlike fully online and fully offline approaches, the *Hybrid* strategy potentially includes both online and offline clusters in the search results. Specifically, *Hybrid* processes a query as follows: (1) uses a search criterion on the offline clusters to select an initial set of candidate clusters for the query results, (2) applies a supervised machine learning classifier on the initial set to identify offline clusters relevant to the query, (3) identifies and re-clusters relevant documents from irrelevant clusters[3] to build online clusters, (4) applies a cluster-ranking criterion to order both offline and online clusters, (5) generates the summaries for the online clusters in the top-k clusters for the query, and (6) returns the top-k clusters.

EXAMPLE 2. Figure 1, which we discussed earlier, showed the summarization-aware results for the query *[Hurricane Wilma]* using our *Hybrid* approach. All clusters in the result are highly relevant to the query. The first and the third clusters are offline clusters classified by the *Hybrid* classifier as relevant to the query. The second cluster was created on the fly and covers the rescue of 250 people from their flooded Cuba homes. This example demonstrates how our approach can (1) reduce the response time by using offline clusters relevant to the query; (2) identify relevant documents from otherwise irrelevant offline clusters; and (3) incorporate relevant online and offline clusters in the query results. □

We now address the main challenges associated with our *Hybrid* approach. Specifically, Section 3.2 discusses the cluster matching and classification steps (steps (1) and (2)). For the document clustering step (step (3)), we use Newsblaster's clustering algorithm, a hierarchical strategy that has been shown to work well for the topical clustering of news articles [3]. Then, Section 3.3 describes the cluster-ranking step (step (4)). Finally, for the multi-document summarization step (step (5)), we use Newsblaster's machine summarizer, a state-of-the-art summarizer that uses different strategies for different types of document clusters [16].

[3]We considered attempting to attach these relevant documents to appropriate relevant clusters from (2), but our experiments showed that this attempt was mostly unsuccessful unless we substantially lowered the similarity thresholds used in the document clustering.

#	*Feature description*
1	Distance of query to cluster's summary
2	Distance of query to a large "document" consisting of the concatenation of all documents in cluster
3	Average distance of query to documents in cluster
4	Average distance of query to matching documents in cluster
5	Distance of query to a "document" consisting of the concatenation of the titles of all documents in cluster
6	Average distance of summary to matching documents in cluster
7	Proportion of matching documents out of all documents in cluster
8	Proportion of matching documents in cluster out of matching documents
9	Proportion of matching documents in cluster out of all documents in matching clusters
10	Proportion of documents in cluster out of all documents in matching clusters
11	Proportion of documents "participating" in summary
12	Proportion of matching documents "participating" in summary
13	Proportion of query terms that appear in cluster summary out of all query terms
14	Proportion of query terms that appear in cluster titles out of all query terms
15	Proportion of query terms that appear, on average, in a cluster title

Table 1: Cluster-level features.

3.2 Cluster Matching and Classification

An essential step of the *Hybrid* strategy is identifying offline clusters relevant to a given query (steps (1) and (2)). For step (1), we retrieve any offline cluster whose summary matches the query or that includes a matching news article. We use a state-of-the-art retrieval model, Okapi [18],[4] for this matching. We address step (2) as a classical classification task. For this, we consider 42 features, described below, to capture the relationship between the query and each offline cluster (e.g., in terms of how well the cluster documents "match" the query). Based on these features, the classifier decides whether the cluster is relevant to the query or not.

Cluster-level Features: In this group (Table 1), features encapsulate properties of each individual cluster. In particular, we consider: (1) the titles of the documents in the cluster; (2) the summary of the cluster; and (3) the text of each document in the cluster. We use Okapi to match the query at hand and the various text components to derive a variety of statistical cluster-based features. Features 1–5 use the distance between the query and different aspects of the cluster documents and summary. Feature 6 relies on the (offline) summary for the cluster and determines how close this summary is to the "good" documents for the query in the cluster. Intuitively, if the cluster summary—which presumably captures the main theme of the cluster—is close to the documents that match the query, then overall we may expect the cluster to be relevant to the query. Features 7–15 capture various statistics on the cluster contents, such as the fraction of documents in the cluster that match the query (Feature 7). In particular, Feature 8 measures the "coverage" of a cluster as the fraction of matching documents in the collection that are part of the cluster. Features 11–13 are summarizer-specific, and focus on the documents in the cluster that contributed text to the summary of the cluster (and are hence, presumably, "central" to the cluster's theme). Among these features, Feature 12 measures the extent to

[4]We also implemented and evaluated our techniques for a Boolean retrieval model, with analogous results.

#	*Feature description*
16-21	Maximum of Features 1 - 6 across all matching clusters
22-27	Minimum of Features 1 - 6 across all matching clusters
28-33	Average of Features 1 - 6 across all matching clusters
34	Proportion of all documents in matching clusters that match the query
35	Proportion of all matching documents out of all documents in collection
36	Proportion of documents in matching clusters out of all documents in collection
37	Average number of cluster documents across all matching clusters
38	Average number of matching documents in cluster across all matching clusters
39-41	Proportion of summaries containing at least one, at least two, and at least three query terms
42	Proportion of matching clusters out of all offline clusters in collection

Table 2: Query-level features.

which the matching documents of a cluster contribute to the cluster summary. As a final example, Feature 13 measures the fraction of query terms that are mentioned in the cluster's summary. (We expect a cluster with a summary that mentions all query terms to be more relevant to the query than a cluster whose summary, say, does not include any query terms.)

Query-level Features: In this group (Table 2), features aggregate their cluster-level counterparts in different ways. Features 16–33 are the maximum, minimum, and average of individual features in Table 1 across all clusters that match the query. Features 34–42 capture various statistics on the content of matching clusters, such as the average number of cluster documents across all matching clusters (Feature 37) and the fraction of summaries of matching clusters that contain query terms (Features 39–41). Other features in this group aim at capturing the "specificity" of a query (e.g., Feature 42 helps distinguish "broad" from "narrow" queries).

Classifier Training: We trained our classifier using *SVM*^{light},[5] an implementation of Support Vector Machines (*SVM*) [21, 9] that has been shown to perform well in text classification and matching problems [4, 8]. We considered different kernels for *SVM* (Section 4.2). We also report results for an efficient rule-based classifier, *Ripper* [1]. We discuss our findings in Section 4.2.

3.3 Cluster Ranking

After the classification step where we identify the document clusters that are relevant to a query (step (2)), our *Hybrid* strategy generates new, query-specific clusters from the relevant articles pulled from irrelevant clusters (step (3)). These clusters are compared during cluster ranking with the relevant offline clusters (step (4)). We now address how to rank the mix of offline and online clusters for a query. This cluster-ranking step is also necessary for the fully online and the fully offline techniques.

Many alternatives have been proposed to rank document clusters for a query (e.g., [5, 11, 7, 19, 14]). We now describe three cluster-ranking functions for our experiments, which capture the most significative alternatives:

Average Okapi Score (AOS): We define the score of cluster c for query q as $Score(c, q) = avg_{d \in c} P(q|d)$, where $P(q|d)$ is the likelihood of generating the query q from document d, computed using the Okapi retrieval model. This strategy is related to a language model for cluster-based retrieval introduced by Liu and Croft [14], where $Score(c, q)$ is defined as the likelihood of generating q from a large "document" that combines all documents in c. Instead of logically concatenating the c documents, we compute the average likelihood of generating q from each individual document in c.

Maximum Okapi Score (MOS): We define $Score(c, q) = \max_{d \in c} P(q|d)$, to pick the "best" cluster document for the query as the cluster representative, under the assumption that the documents in a cluster are sufficiently similar to each other.

Distance from Centroid (DC): We define $Score(c, q) = -1 \cdot Distance(Centroid(c), d)$, where $d = \arg\max_{d_i \in c} P(q|d_i)$ and $Centroid(c) = \frac{1}{|c|} \times \sum_{d_i \in c} \overrightarrow{d_i}$. Again, we use the Okapi model to compute $Distance$ and $P(q|d_i)$, as well as the vector representation $\overrightarrow{d_i}$ of document d_i. This strategy attempts to measure the distance between a cluster's centroid and the user query using the document with the highest Okapi score. A centroid close to the most relevant document might indicate high overall relevance of the cluster to the query. This strategy is similar to using the cluster centroid or summary to determine the cluster rank [25, 24].

Many additional alternative ranking strategies are possible, of course. In addition to the three options above, for which we report experimental results, we also experimented with other alternatives, including defining the score of a cluster as the minimum Okapi score among the documents in the cluster with non-zero score, and others. None of these additional strategies worked well in our environment, so we do not discuss them further.

4. EXPERIMENTAL EVALUATION

We now describe our data and queries (Section 4.1), and the evaluation results for our *Hybrid* classifier (Section 4.2) and search techniques (Section 4.3).

4.1 Data Collection and Queries

For training and testing, we rely on news articles crawled and processed by Newsblaster. Additionally, we conducted user studies to collect queries and their associated relevance judgments for our experiments. Unfortunately, we could not rely for our experiments on a less-expensive evaluation involving standard test collections such as the TREC ad-hoc test collections, because they do not include the variety of alternative news sources for every event that we expect in news portals, and such variety is critical to make the kind of multi-document summarization on which we focus in this paper meaningful.

Document Collection: Our document collection consists of six daily Newsblaster runs, and includes 1,620 clusters of 13,017 stories gathered from 18 U.S. news sources, together with the Newsblaster summaries for the clusters. Each day's run contains (1) the news documents crawled that day, (2) the Newsblaster clusters and their machine summaries, and (3) all metadata discovered, such as keywords, clustering hierarchy, and cluster classification.

Queries: Our query set consists of 184 queries submitted over our document collection by 22 users, mostly PhD students and researchers working on NLP and IR, as well as a few journalists. We randomly split the queries into three sets $Q1$, $Q2$, and $Q3$, with 73, 77, and 34 queries, respectively. For searching and indexing the documents, we used the Okapi retrieval model as implemented in Lemur,[6] an open-source toolkit designed to facilitate research in language modeling and information retrieval, and, as an alternative, the Boolean retrieval model as implemented in *Jakarta Lucene*,[7] an open-source full-text search engine.

Cluster-level Relevance Judgments: Based on an adaptation of

[5] svmlight.joachims.org

[6] www.lemurproject.org

[7] lucene.apache.org

TREC guidelines,[8] participants manually labeled each offline cluster as relevant or irrelevant for each query in $Q1$ and $Q2$. Users from the same group who issued the queries for our experiments participated in the cluster labeling. To decide if a cluster is relevant to a query, we allowed the labelers to only examine its associated summary and the titles of its documents.

Document-level Relevance Judgments: Following similar guidelines for relevance as for the clusters, participants manually labeled a pool of documents as relevant or irrelevant for the $Q3$ queries. Specifically, for each query we asked our human raters to label the top-20 documents returned by each search alternative that we compare in Section 4.3, for a total of up to 66 documents per query. For the cluster-based techniques, the top-20 documents are obtained by a "linearization" of the top clusters [5, 14, 23], where we replace each cluster with a ranked list of its documents[9]. We assigned three different raters for each query and computed the relevance of each document as the majority vote of the three raters. In total, 52 users (mostly English native speakers, including 38 PhD students from different backgrounds, eight journalists, one undergraduate student, three sales persons, and two software engineers) labeled 102 pools of up to 66 stories each. Raters for the same query were forced into three different orders of the documents associated with the query, to eliminate any order-introduced bias. We analyzed the labeling by investigating the interrater (assessor) agreement, using the Kappa [2] measure of agreement among our raters to identify potential problems. More than 75 percent of the queries have a "very good" Kappa strength while only four queries have "fair" strength. We examined all judgments of these four queries and concluded that it was natural to have such agreement strength between the raters due to the differences among raters' contexts and conceptions. For instance, a historian rated most of the news stories about President Bush's visit to Liberia as relevant to the query *[Liberia history]* while another rater, a journalist, insisted that since the visit took place at the time of the query, then the news stories cannot be considered as history yet.

4.2 Cluster Classifier

We now evaluate the cluster classifier of Section 3.2, which is trained once and for all using the $Q1$ training set. Later, an unseen collection of clusters and a new query are processed by first calculating the values of the features in Tables 1 and 2; then, the classifier decides on the relevance of each cluster for the query based on the feature values.

Settings: As training and test sets, we used the cluster-level relevance judgments for the $Q1$ and $Q2$ query sets, respectively. Each example in these sets corresponds to a pair $<q, c>$, where q is a query and c is an offline document cluster, and is represented using the features described in Section 3.2. Finally, each $<q, c>$ example is labeled (see Section 4.1), indicating whether cluster c is relevant for query q or not. We measure classification accuracy as the average fraction of correctly classified clusters per query (***Q-Accuracy***), which we define as $\frac{1}{|Q|} \times \sum_{q \in Q} \frac{|\hat{C}_q|}{|C_q|}$, where C_q is the set of "candidate" clusters for q and $\hat{C}_q$ is the set of clusters that are correctly classified as relevant or not for q.

Techniques for Comparison: We used *SVM* and *Ripper* to train our classifier (Section 3.2). For *SVM*, we experimented with linear, polynomial, and radial basis kernels. In addition, we implemented a baseline learner, which uses the average distance of a query to the documents in a cluster (Feature 3) to decide cluster relevance to the query and learns the value of this feature that obtains the best classification performance on the training set[10].

Results of Feature Selection Step: We used forward and backward feature selection algorithms [6, 10] provided in the machine learning toolkit YALE[11]. We also adapted the latter to *SVM* with linear kernels by discarding, in each iteration, features whose weight is close to zero. We performed a feature selection step over the training query set $Q1$, for the different classifiers that we tried. In general, we found that features based on simple counting such as Feature 15 are less useful than the powerful Okapi-based features such as Feature 1 (Table 1). Table 3 shows an example "winning" feature set for *SVM* with linear kernels. In general, the winning features include both query-level and cluster-level features, as well as features based on the Okapi model. We also find that Features 7 and 8 appear consistently in the majority of the winning sets. Intuitively, to determine the relevance of a cluster for a query, we need to examine both the properties of the cluster as well as the general nature of the query, as revealed by the query-level features. This can also be noticed in *Ripper's* classification rules, where both cluster- and query-level features are used in the prediction.

Feature Type	*Features*
Cluster-level	1, 2, 4, 5, 7, 8, 12, 14
Query-level	17, 18, 23, 25, 31, 42

Table 3: A set of "winning" classifier features.

Learner	Q-Accuracy	
	Training ($Q1$)	*Test ($Q2$)*
SVM linear	90.5%	**88.4%**
SVM polynomial	91.12%	85.88%
SVM radial basis	89.06%	86.89%
Ripper	85.7%	86.8%
Baseline	84%	81.4%
AlwaysIrrelevant	65.64%	66.92%

Table 4: Cluster classification accuracy.

Classification Accuracy: Table 4 shows the accuracy of the classifiers learned using *Ripper*, *SVM*, and the baseline. As a sanity check, we also include a line, *AlwaysIrrelevant*, for a technique that always guesses "irrelevant." *SVM* and *Ripper* have high accuracy and outperform the baseline. For the search quality evaluation in the next section, we choose the features in Table 3 and *SVM* with linear kernels.

4.3 Search Alternatives

We now report our evaluation of the search alternatives using the $Q3$ query set.

Techniques for Comparison: As a baseline, we consider the fully offline approach of Section 2.1, which returns static, offline clusters and their summaries. We study two fully offline variants using the Okapi retrieval model to match the queries against the documents, summaries, and other cluster components, as appropriate: *OffDocOkapi* uses the documents to retrieve the candidate clusters; *OffSumOkapi*, instead, only relies on the summaries. As another baseline, we consider the fully online approach of Section 2.2, which performs online clustering and summarizes the articles that match

[8] trec.nist.gov/data/reljudge_eng.html

[9] For this intra-cluster document ranking, we first rank the matching documents in a cluster using the retrieval model associated with the technique in question, and then add the non-matching documents in random order.

[10] We also tried other variations of this baseline with different individual features and did not find substantial performance differences between them.

[11] yale.cs.uni-dortmund.de

Name	*Approach*
OffDocOkapi	*Offline*, choosing clusters by matching documents and query
OffSumOkapi	*Offline*, choosing clusters by matching summaries and query
OnOkapi	*Online*
HybridOkapi	*Hybrid*
FlatOkapi	*Document* ranking using Okapi

Table 5: Okapi-based search techniques.

Name	*Approach*
OffDocBoolean	*Offline*, choosing clusters by matching documents and query
OffDSumBoolean	*Offline*, choosing clusters by either matching documents and query or matching summaries and query
OnBoolean	*Online*
HybridBoolean	*Hybrid*
FlatBoolean	*Document* ranking using Boolean

Table 6: Boolean-based search techniques.

a query at query-execution time. We refer to the version of this baseline that uses Okapi as *OnOkapi*. We also evaluate our *Hybrid* strategy, which potentially returns both online and offline clusters, and refer to the version that uses Okapi as *HybridOkapi*. We thus compared four cluster-based search techniques, *OffDocOkapi*, *OffSumOkapi*, *OnOkapi*, and *HybridOkapi* (first four lines of Table 5), each with the three cluster-ranking methods described in Section 3.3 and using the Okapi retrieval model to match the queries against the documents, summaries, and other cluster components. Analogously, we study another set of techniques (first four lines of Table 6) that use the conjunctive Boolean retrieval model, namely *OffDocBoolean*, *OffDSumBoolean*, *OnBoolean*, and *HybridBoolean*. Since the conjunctive Boolean retrieval model is strict, *OffDSumBoolean* returns clusters with matching summaries or *articles*, unlike the analogous Okapi technique *OffSumOkapi*. As mentioned, our focus in this paper is on search techniques that return document clusters with their multi-document summaries; however –and as a sanity check– we also compare our techniques against two regular *flat* document-based result ranking techniques: *FlatOkapi* (last line of Table 5), which uses the Okapi retrieval model, and *FlatBoolean* (last line of Table 6), which uses the Boolean model.

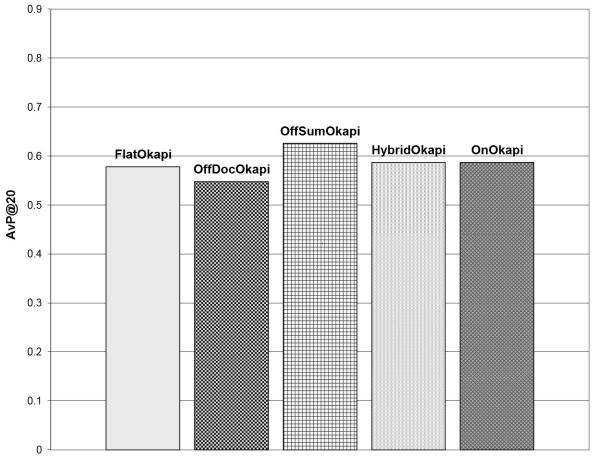
Evaluation Metrics: To evaluate the quality of the search output of the various alternative techniques, we use document-level relevance judgments, which allow us to use the well-established document-level precision metric to compare our techniques. Since efficiency is a major motivation behind our work, we also compute the response time of each technique.

Precision at k: We compute the precision for the first k documents in each query result, where the cluster-based results are linearized as in Section 4.1. We report the average precision at 20, AvP@20[12].

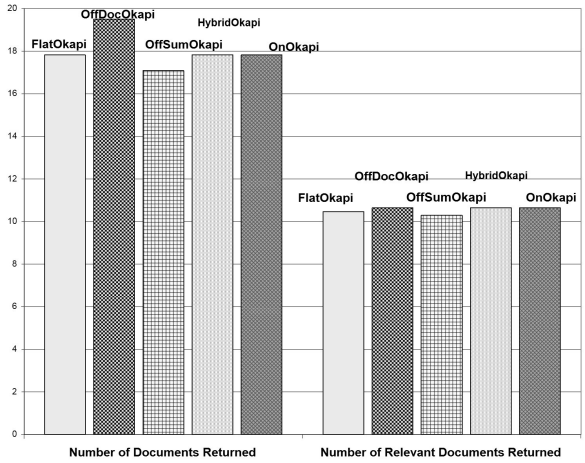
Response Time: We compute the time for each technique to return the query results, averaged over five runs. For each run and query, we restart all different components of the search system to eliminate any effect related to caching from previous runs. Finally, we only summarize new, online clusters if needed to return 20 documents overall for a query.

Selecting the Best Cluster-Ranking Schema: We evaluated each cluster-based technique with our three cluster-ranking schemes by measuring AvP@20. Both *MOS* and *AOS* (Section 3.3) always outperform *DC* with a slight advantage of *MOS*. We use *MOS* for the experiments that we report next.

[12]We also computed AvP@10 and obtained similar results.



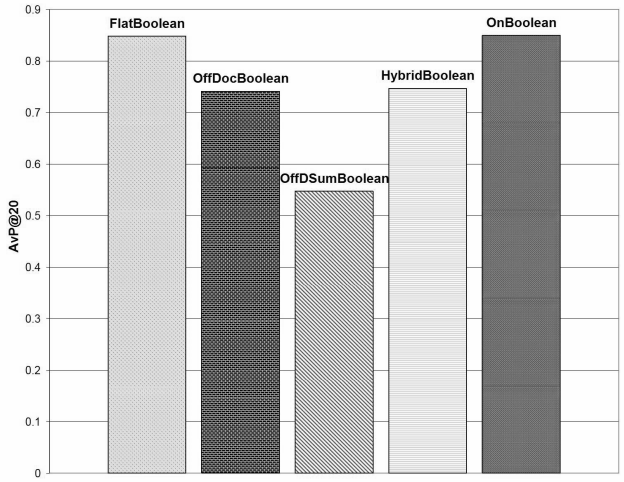
(a) AvP@20 for $Q3$ queries.



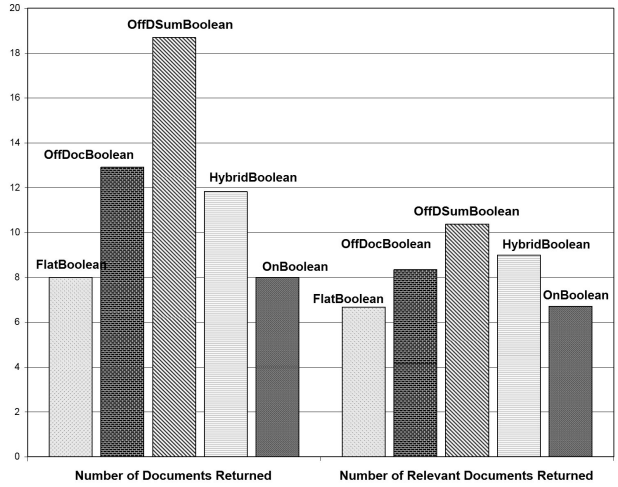
(b) Average number of documents retrieved overall and of *relevant* documents returned for $Q3$ queries.

Figure 2: Okapi-based search with *MOS* ranking.

Accuracy of Okapi-Based Techniques (Table 5): Figure 2 (a) shows that *OnOkapi*, *HybridOkapi*, and *FlatOkapi* have similar AvP@20, with values of 0.587, 0.588, and 0.578, respectively. *HybridOkapi* maintains a precision similar to *OnOkapi* (which is prohibitively expensive) and *FlatOkapi* (which does not return document clusters and summaries). Furthermore, the three techniques return a similar number of relevant documents (about 10 documents on average per query) and a similar number of documents overall in results (17.8 documents). *OffSumOkapi* has the highest AvP@20, 0.625, which we believe can be explained by a slight bias introduced by our query collection procedure: specifically, note that our human subjects defined the queries for our experiments by browsing over Newsblaster cluster summaries. (See Section 6.) Figure 2 (b) shows that all techniques return, on average, about 10 relevant documents per query, and at least 17 documents overall. The static *OffDocOkapi* has the lowest precision and returns about two more irrelevant documents than any other technique due to the non-matching documents in the static clusters.

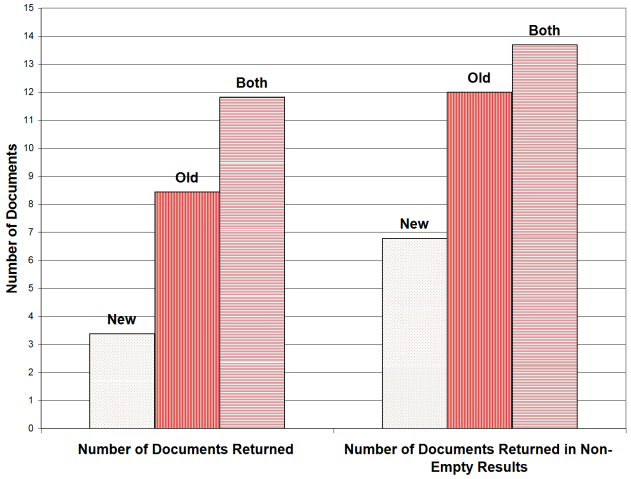


(a) AvP@20 for $Q3$ queries.

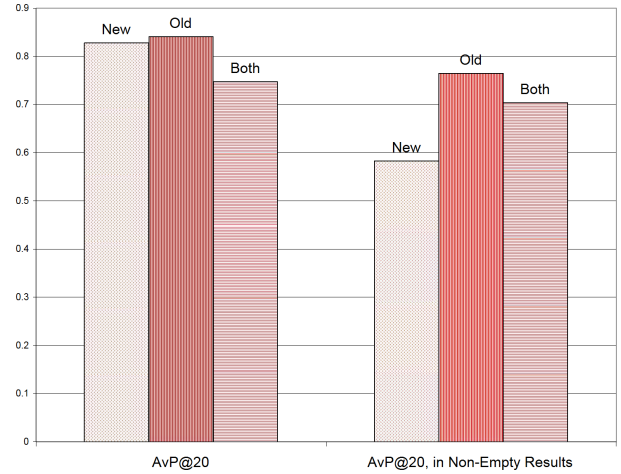


(b) Average number of documents retrieved overall and of *relevant* documents returned for $Q3$ queries.

Figure 3: Boolean-based search with *MOS* ranking.



(a) Distribution of documents across "old" and "new" clusters.

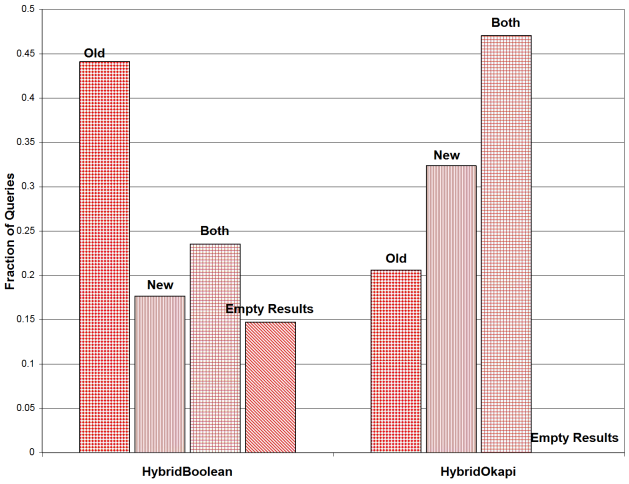


(b) Distribution of AvP@20 across "old" and "new" clusters.

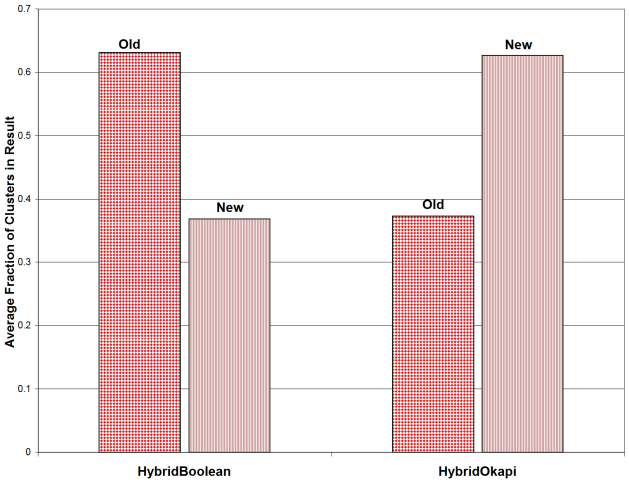
Figure 4: Characteristics of "old" and "new" clusters in the *HybridBoolean* technique results.

Accuracy of Boolean-Based Techniques (Table 6): Figure 3 (a) shows that *FlatBoolean* and *OnBoolean* exhibit the highest AvP@20, with values of 0.848 and 0.849, respectively. *OffDocBoolean* and *HybridBoolean* are a relatively close second, with an AvP@20 of 0.741 and 0.747, respectively. Unfortunately, the high precision of *FlatBoolean* and *OnBoolean* is due to the fact that these techniques produce a high percentage of empty results, because of the strict nature of the conjunctive Boolean retrieval model. To reveal this problem, Figure 3 (b) shows the average number of *relevant* documents returned by each technique, as well as the average number of documents returned overall. Both *FlatBoolean* and *OnBoolean* return, on average, only eight documents per query. (These techniques only return matching documents.) In contrast, the techniques that match entire clusters to the query, namely *OffDocBoolean*, *OffDSumBoolean*, and *HybridBoolean*, return more documents in the query results, with *OffDocBoolean* and *HybridBoolean* achieving the best precision among them. Interestingly, *OffDocBoolean's* AvP@20 decreases to 0.65 when we disregard queries with empty results, while it is 0.7 for *HybridBoolean*. Also, *HybridBoolean* outperforms *FlatBoolean* for most of the queries in $Q3$. The differences are statistically significant as determined by a paired t-test [15], and indicate the benefits that *HybridBoolean* gains from the inclusion of non-matching but relevant articles.

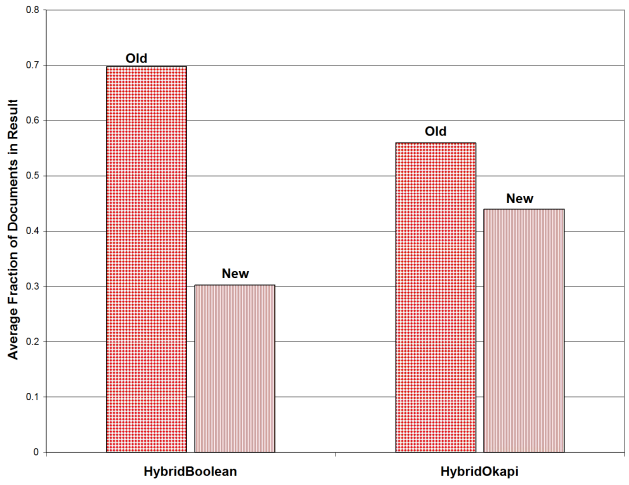
Use of Offline Clusters: To further investigate the behavior of the *Hybrid* approach, we examine its use of static clusters. The results show that *HybridBoolean* uses the static clusters intensively but selectively: Figure 4 (a) shows that among the top-20 returned documents there are, on average, 8.5 documents originated in 1.5 "old" (offline) clusters. Furthermore, Figure 4 (b) shows a high AvP@20 for the *HybridBoolean* technique among the "old" (offline) and "new" (online) clusters separately, with values of 0.84 and 0.82, respectively. Moreover, Figure 5 (a) shows that about 45% of the queries returned only "old" clusters and about 23% returned both types of clusters for *HybridBoolean*. In addition, Figure 5 (b) shows that, on average, 63% of the returned clusters are "old" clusters for *HybridBoolean*. Figure 5 (c) shows that 70% of the returned documents originated in "old" clusters for *HybridBoolean*. These results show that our methodology selectively uses static clusters and that this does not hurt the overall precision of the *HybridBoolean* technique. Similar conclusions can be drawn for



(a) Fraction of queries that returned "old", "new", and both types of clusters.



(b) Fraction of "old" and "new" clusters in result.



(c) Fraction of documents from "old" and "new" clusters in result.

Figure 5: Characteristics of query results for the *Hybrid* techniques.

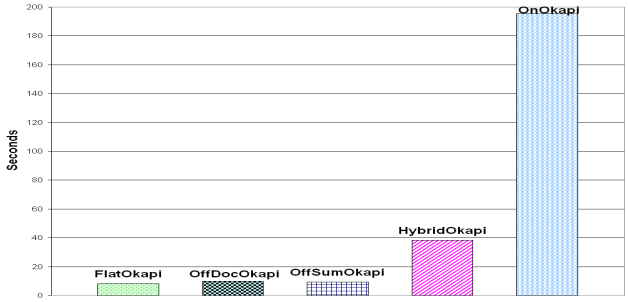


Figure 6: Average response time for $Q3$ queries.

the *HybridOkapi* technique, with the clarification that the increase in the fraction of "new" returned clusters (Figure 5 (b)) is due to singleton clusters. The fraction of the returned documents originated in "old" clusters for *HybridOkapi* is still high (Figure 5 (c)). These results highlight the role of the *Hybrid* classifier in identifying relevant offline clusters, which in turn reduces the response time of the technique, as we discuss next.

Efficiency: A main aim of this paper is to reduce the response time in summarization-aware search. Figure 6 shows the response time for the techniques. Not surprisingly, *OnOkapi* has the largest response time, with 195 seconds on average per query. *HybridOkapi*, with similar precision, has a response time of only 38 seconds. As predicted, the offline techniques are the least expensive ones, since they do not perform clustering or summarization on the fly.

Evaluation Summary: As a general conclusion, the offline cluster-based techniques are attractive candidates for scenarios where response time is more important than having query-specific document clusters and summaries. On the other hand, online cluster-based search techniques have high precision, at the expense of unacceptable query-execution times for these techniques. Interestingly, *Hybrid* always performed at least as well as a state-of-the-art document-based approach in terms of result quality, so a careful use of offline clusters does not damage the overall result accuracy. On the contrary, offline clusters, introduced by *Hybrid*, are useful for identifying relevant documents and reducing the cost of summarization at query-execution time.

5. RELATED WORK

Document summarization has attracted substantial attention in the NLP community for many years and multi-document summarization systems have started to emerge (e.g., [12, 16, 17]). We use Newsblaster, a state-of-the-art multi-document summarization system that uses different strategies for different types of document clusters, for our work on summarization-aware search. Besides the quality of the summaries, a main concern in NLP, the efficiency of creating such summaries is a major motivation for our work. Our approach is largely independent of the summarization system of choice, and we can easily incorporate advances in multi-document summarization as they happen.

In this paper, we study query processing techniques where the query results are document clusters, with their corresponding summaries. Document clustering has been extensively used in IR applications to improve retrieval efficiency [22] and effectiveness [7]. Over the years, many cluster-based search methods, whether returning static, offline clusters or creating query-specific, online clusters on the fly, have been introduced and evaluated [5, 11, 7, 19, 23].

Figure 7: Our summarization-aware search strategies over the Newsblaster news portal.

Several cluster-based search engines[13] for the web have emerged. Recently, Liu and Croft [14] introduced a promising cluster-based retrieval approach based on language modeling.

Hearst and Pedersen [5], among others, have claimed that query-specific, cluster-based search results exhibit good precision, while Tombros et al. [19] have showed that online clustering of query results significantly outperformed offline cluster-based retrieval. Unfortunately, state-of-the-art multi-document summarizers are slow, since they rely heavily on inherently slow NLP tools for parsing, tagging, and discourse and sentence analysis [13], which compromises the applicability of online summarization-aware search approaches in practice for efficiency considerations. Close to this fully online approach, NewsInEssence [17] generates online document clusters and summaries for user-specified information needs. In Section 4.3, we compared our *Hybrid* technique experimentally against an online approach that is similar in spirit to NewsInEssence.

6. CONCLUSION AND FUTURE WORK

We investigated a family of summarization-aware search techniques, where the query results consist of document clusters together with their machine-generated multi-document summaries. Our *Hybrid* search strategy addresses the limitations of fully online –slow execution due to the need for on-the-fly summarization– and fully offline –query-independent document clusters and summaries– summarization-aware search approaches. Our extensive experiments show that our strategy exploits, when possible, offline clusters and their associated summaries to reduce the cost of summarization at query-execution time, at the same time achieving high query-result precision. An additional conclusion is that the offline cluster-based techniques might be attractive candidates for scenarios where response time is more important than having query-specific document clusters and summaries. As a final observation, our strategy could be "tuned" to achieve the desired balance between efficiency and query-specificity of the document clusters and summaries produced, by appropriately biasing the cluster classifier on which we rely.

We have fully deployed our summarization-aware search strategies as part of Newsblaster, and made them publicly accessible at newsblaster.cs.columbia.edu[14]. Users can choose among the Okapi-based search techniques *OffSumOkapi*, *OffDocOkapi*, *HybridOkapi*, and *OnOkapi*, with the *MOS* ranking scheme, to obtain summarization-aware search results. Figure 7 is a screenshot of the Newsblaster interface, showing a search box and an associated pull-down menu with the search-technique options on the top left corner.

As future work, we are planning to expand our evaluation in two directions. First, we will include real-user queries, to avoid the evaluation bias that we discussed in Section 4.3. Second, we will conduct user studies to measure the relevance to a query of the multi-document summaries in the query results. Intuitively, we expect to confirm experimentally that the query-specific summaries in the fully online and *Hybrid* techniques are indeed more relevant to the query in question than the query-independent fully offline summaries. As a somewhat less clear question, we will also study whether the *Hybrid* summaries—which require modest computational resources to generate—are comparable in relevance to the fully online summaries—which are prohibitively expensive to derive.

[13]Examples include www.clusty.com and www.vivisimo.com.

[14]Due to copyright-related issues, Newsblaster is now password-protected.

Acknowledgments

This work was supported in part by the National Science Foundation under the KDD program and in part by a research gift from Microsoft Research. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of either NSF or Microsoft. We also thank Sasha Blair-Goldensohn, Silviu-Petru Cucerzan, and Panagiotis G. Ipeirotis for their helpful comments. Finally, we express our gratitude to the students from Columbia University and the National University of Singapore who patiently participated in our human studies.

7. REFERENCES

[1] W. W. Cohen. Fast effective rule induction. In *ICML'95*, 1995.

[2] J. L. Fleiss, B. Levin, M. C. Paik, J. Fleiss, and B. Levin. *Statistical Methods for Rates Proportions*. Wiley-Interscience, 2003.

[3] V. Hatzivassiloglou, L. Gravano, and A. Maganti. An investigation of linguistic features and clustering algorithms for topical document clustering. In *SIGIR 2000*, 2000.

[4] M. A. Hearst. Trends and controversies: Support vector machines. *IEEE Intelligent Systems*, 13(4), July 1998.

[5] M. A. Hearst and J. O. Pedersen. Reexamining the cluster hypothesis: Scatter/gather on retrieval results. In *SIGIR'96*, 1996.

[6] A. Jain and D. Zongker. Feature selection: evaluation, application and small sample performance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(2), Feb. 1997.

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

[8] T. Joachims. Making large-scale support vector machine learning practical. In *Advances in Kernel Methods - Support Vector Learning*. MIT Press, 1999.

[9] T. Joachims. *Learning to Classify Text Using Support Vector Machines: Methods, Theory, and Algorithms*. Kluwer Academic Publishers, 2002.

[10] D. Koller and M. Sahami. Toward optimal feature selection. In *ICML'96*, 1996.

[11] A. Leuski and J. Allan. Improving interactive retrieval by combining ranked list and clustering. In *RIAO'00*, 2000.

[12] C.-Y. Lin and E. Hovy. Automated multi-document summarization in NeATS. In *HLT'02*, 2002.

[13] K. C. Litkowski. Summarization experiments in DUC 2004. In *DUC'04*, 2001.

[14] X. Liu and B. W. Croft. Cluster-based retrieval using language models. In *SIGIR 2004*, 2004.

[15] J. P. Marques De Sá. *Applied Statistics*. Springer Verlag, 2003.

[16] K. R. McKeown et al. Tracking and summarizing news on a daily basis with Columbia's Newsblaster. In *HLT'02*, 2002.

[17] D. R. Radev et al. NewsInEssence: A system for domain-independent, real-time news clustering and multi-document summarization. In *HLT'01*, 2001.

[18] S. E. Robertson. Overview of the Okapi projects. *Journal of Documentation*, 53(1):3–7, 1997.

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

[20] C. J. van Rijsbergen. *Information Retrieval*. Butterworths, 1979.

[21] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer Verlag, 1995.

[22] E. M. Voorhees. *The effectiveness and efficiency of agglomerative hierarchic clustering in document retrieval*. PhD thesis, Cornell University, 1985.

[23] O. Zamir and O. Etzioni. Web document clustering: A feasibility demonstration. In *SIGIR'98*, 1998.

[24] O. Zamir and O. Etzioni. Grouper: a dynamic clustering interface to web search results. In *WWW8*, 1999.

[25] H.-J. Zeng, Q.-C. He, Z. Chen, W.-Y. Ma, and J. Ma. Learning to cluster web search results. In *SIGIR 2004*, 2004.