Cheshire II at INEX '04: Fusion and Feedback for the Adhoc and Heterogeneous Tracks

Ray R. Larson

School of Information Management and Systems,
University of California, Berkeley, California, USA, 94720-4600
ray@sims.berkeley.edu

Abstract. This paper describes the retrieval approach used by UC Berkeley in the adhoc and heterogeneous tracks for the 2004 INEX evaluation. As in previous INEX evaluations, the main technique we are testing is the fusion of multiple probabilistic searches against different XML components using both Logistic Regression (LR) algorithms and a version of the Okapi BM-25 algorithm in conjunction with Boolean constraints for some elements. We also describe some additional experiments, subsequent to INEX that promise further improvements in results.

1 Introduction

In this paper we describe the document and component fusion approaches used by U.C. Berkeley for INEX 2004, the results from our official INEX submissions, and the results of subsequent analysis and re-testing. This work is based upon and extends the work described in the special INEX issue of the Journal of Information Retrieval[9]. In addition we will discuss the approach taken for INEX Heterogeneous track.

The basic approach that we use for the INEX 2004 retrieval tasks is based on some early work done in TREC, where it was found that fusion of multiple retrieval algorithms provided an improvement over a single search algorithm[14, 2]. Later analyses of these fusion approaches[10, 1] indicated that the greatest effectiveness improvements appeared to occur between relatively ineffective individual methods, and the fusion of ineffective techniques, while often approaching the effectiveness of the best single IR algorithms, seldom exceeded them for individual queries and never exceeded their average performance. In our analysis of fusion approaches for XML retrieval[9], based on runs conducted after the 2003 INEX meeting, we conducted analyses of the overlap between result sets across algorithm and also examined the contributions of different XML document components to the results.

The remainder of the paper is organized as follows: we will first discuss the algorithms and fusion operators used in our official INEX 2004 adhoc runs and for the heterogeneous track. Then we will look at how these algorithms and operators were used in the various submissions for the adhoc and heterogeneous tracks, and finally we will examine the results and discuss directions for future research.

2 The Retrieval Algorithms and Fusion Operators

In [9] we conducted an analysis of the overlap between the result lists retrieved by our Logistic Regression algorithm and the Okapi BM-25 algorithm. We found that, on average, over half of the result lists retrieved by each algorithm in these overlap tests were both non-relevant *and* unique to that algorithm, fulfilling the main criteria for effective algorithm combination suggested by Lee[10]: that the algorithms have similar sets of relevant documents and different sets of non-relevant. This section is largely a repetition of the material presented in [9] with additional discussion of the re-estimation of the LR parameters for different XML components and indexes used in the INEX 2004 tests.

In the remainder of this section we describe the Logistic Regression and Okapi BM-25 algorithms that were used for the evaluation and we also discuss the methods used to combine the results of the different algorithms. The algorithms and combination methods are implemented as part of the Cheshire II XML/SGML search engine [7, 8, 6] which also supports a number of other algorithms for distributed search and operators for merging result lists from ranked or Boolean sub-queries. Finally, we will discuss the re-estimation of the LR parameters for a variety of XML components of the INEX test collection.

2.1 Logistic Regression Algorithm

The basic form and variables of the *Logistic Regression* (LR) algorithm used was originally developed by Cooper, et al. [4]. It provided good full-text retrieval performance in the TREC ad hoc task and in TREC interactive tasks [5] and for distributed IR [6]. As originally formulated, the LR model of probabilistic IR attempts to estimate the probability of relevance for each document based on a set of statistics about a document collection and a set of queries in combination with a set of weighting coefficients for those statistics. The statistics to be used and the values of the coefficients are obtained from regression analysis of a sample of a collection (or similar test collection) for some set of queries where relevance and non-relevance has been determined. More formally, given a particular query and a particular document in a collection $P(R \mid Q, D)$ is calculated and the documents or components are presented to the user ranked in order of decreasing values of that probability. To avoid invalid probability values, the usual calculation of $P(R \mid Q, D)$ uses the "log odds" of relevance given a set of S statistics, s_i, derived from the query and database, such that:

$$\log O(R \mid Q, D) = b_0 + \sum_{i=1}^{S} b_i s_i \tag{1}$$

where b_0 is the intercept term and the b_i are the coefficients obtained from the regression analysis of the sample collection and relevance judgements. The final ranking is determined by the conversion of the log odds form to probabilities:

$$P(R \mid Q, D) = \frac{e^{\log O(R|Q,D)}}{1 + e^{\log O(R|Q,D)}} \tag{2}$$

Based on the structure of XML documents as a tree of XML elements, we define a “document component” as an XML subtree that may include zero or more subordinate XML elements or subtrees with text as the leaf nodes of the tree. For example, in the XML Document Type Definition (DTD) for the INEX test collection defines an article (marked by XML tag <*article*>) that contains front matter (<*fm*>), a body (<*bdy*>) and optional back matter (<*bm*>). The front matter (<*fm*>), in turn, can contain a header <*hdr*> and may include editor information (<*edinfo*>), author information (<*au*>), a title group (<*tig*>), abstract (<*abs*>) and other elements. A title group can contain elements including article title (<*atl*>) the page range for the article (<*pn*>), and these in turn may contain other elements, down to the level of individual formatted words or characters. Thus, a component might be defined using any of these tagged elements. However, *not all possible components are likely to be useful* in content-oriented retrieval (e.g., tags indicating that a word in the title should be in italic type, or the page number range) therefore we defined the retrievable components selectively, including document sections and paragraphs from the article body, and bibliography entries from the back matter (see Table 3).

Naturally, a full XML document may also be considered a “document component”. As discussed below, the indexing and retrieval methods used in this research take into account a selected set of document components for generating the statistics used in the search process and for extraction of the parts of a document to be returned in response to a query. Because we are dealing with not only full documents, but also document components (such as sections and paragraphs or similar structures) derived from the documents, we will use C to represent document components in place of D. Therefore, the full equation describing the LR algorithm used in these experiments is:

$$\begin{aligned}\log O(R \mid Q, C) = {} & \\ & b_0 + \left(b_1 \cdot \left(\frac{1}{|Q_c|}\sum_{j=1}^{|Q_c|} \log qtf_j\right)\right) \\ & + \left(b_2 \cdot \sqrt{|Q|}\right) \\ & + \left(b_3 \cdot \left(\frac{1}{|Q_c|}\sum_{j=1}^{|Q_c|} \log tf_j\right)\right) \\ & + \left(b_4 \cdot \sqrt{cl}\right) \\ & + \left(b_5 \cdot \left(\frac{1}{|Q_c|}\sum_{j=1}^{|Q_c|} \log \frac{N - n_{t_j}}{n_{t_j}}\right)\right) \\ & + \left(b_6 \cdot \log |Q_d|\right)\end{aligned} \tag{3}$$

Where:
Q is a query containing terms T,
$|Q|$ is the total number of terms in Q,

$|Q_c|$ is the number of terms in Q that also occur in the document component,
tf_j is the frequency of the jth term in a specific document component,
qtf_j is the frequency of the jth term in Q,
n_{t_j} is the number of components (of a given type) containing the jth term,
cl is the document component length measured in bytes.
N is the number of components of a given type in the collection.
b_i are the coefficients obtained though the regression analysis.

This equation, used in estimating the probability of relevance in this research, is essentially the same as that used in [3]. The b_i coefficients in the "Base" version of this algorithm were estimated using relevance judgements and statistics from the TREC/TIPSTER test collection. In INEX 2004 we used both this Base version and a version where the coeffients for each of the major document components were estimated separately and combined through component fusion. The coefficients for the Base version were $b_0 = -3.70, b_1 = 1.269, b_2 = -0.310, b_3 = 0.679, b_4 = -0.021, b_5 = 0.223$ and $b_6 = 4.01$. We will discuss the re-estimated coefficients for the various document components and indexes later in this section.

2.2 Okapi BM-25 Algorithm

The version of the Okapi BM-25 algorithm used in these experiments is based on the description of the algorithm in Robertson [12], and in TREC notebook proceedings [13]. As with the LR algorithm, we have adapted the Okapi BM-25 algorithm to deal with document components :

$$\sum_{j=1}^{|Q_c|} w^{(1)} \frac{(k_1+1)tf_j}{K+tf_j} \frac{(k_3+1)qtf_j}{k_3+qtf_j} \tag{4}$$

Where (in addition to the variables already defined):
K is $k_1((1-b)+b \cdot dl/avcl)$
k_1**,** b **and** k_3 are parameters (1.5, 0.45 and 500, respectively, were used),
$avcl$ is the average component length measured in bytes
$w^{(1)}$ is the Robertson-Sparck Jones weight:

$$w^{(1)} = \log \frac{\left(\frac{r+0.5}{R-r+0.5}\right)}{\left(\frac{n_{t_j}-r+0.5}{N-n_{t_j}-R-r+0.5}\right)}$$

r is the number of relevant components of a given type that contain a given term,
R is the total number of relevant components of a given type for the query.

Our current implementation uses only the *a priori* version (i.e., without relevance information) of the Robertson-Sparck Jones weights, and therefore the $w^{(1)}$ value is effectively just an IDF weighting. The results of searches using our implementation of Okapi BM-25 and the LR algorithm seemed sufficiently different to offer the kind of conditions where data fusion has been shown to be be most effective [10], and our overlap analysis of results for each algorithm (described in the evaluation and discussion section) has confirmed this difference and the fit to the conditions for effective fusion of results.

2.3 Boolean Operators

The system used supports searches combining probabilistic and (strict) Boolean elements, as well as operators to support various merging operations for both types of intermediate result sets. Although strict Boolean operators and probabilistic searches are implemented within a single process, using the same inverted file structures, they really function as two parallel *logical* search engines. Each logical search engine produces a set of retrieved documents. When a only one type of search strategy is used then the result is either a probabilistically ranked set or an unranked Boolean result set. When both are used within in a single query, combined probabilistic and Boolean search results are evaluated using the assumption that the Boolean retrieved set has an estimated $P(R \mid Q_{bool}, C) = 1.0$ for each document component in the set, and 0 for the rest of the collection. The final estimate for the probability of relevance used for ranking the results of a search combining strict Boolean and probabilistic strategies is simply:

$$P(R \mid Q, C) = P(R \mid Q_{bool}, C) P(R \mid Q_{prob}, C) \tag{5}$$

where $P(R \mid Q_{prob}, C)$ is the probability of relevance estimate from the probabilistic part of the search, and $P(R \mid Q_{bool}, C)$ is the Boolean. In practice the combination of strict Boolean "AND" and the probablistic approaches has the effect of restricting the results to those items that match the Boolean part, with ranking based on the probabilistic part. Boolean "NOT" provides a similar restriction of the probabilistic set by removing those document components that match the Boolean specification. When Boolean "OR" is used the probabilistic and Boolean results are merged (however, items that only occur in the Boolean result, and not both, are reweighted as in the "fuzzy" and merger operations described below.

A special case of Boolean operators in Cheshire II is that of proximity and phrase matching operations. In proximity and phrase matching the matching terms must also satisfy proximity constraints (both term order and adjacency in the case of phrases). Thus, proximity operations also result in Boolean intermediate result sets.

2.4 Result Combination Operators

The Cheshire II system used in this evaluation provides a number of operators to combine the intermediate results of a search from different components or indexes. With these operators we have available an entire spectrum of combination methods ranging from strict Boolean operations to fuzzy Boolean and normalized score combinations for probabilistic and Boolean results. These operators are the means available for performing fusion operations between the results for different retrieval algorithms and the search results from different different components of a document. We will only describe one of these operators here, because it was the only type used in the evaluation reported in this paper.

The MERGE_CMBZ operator is based on the "CombMNZ" fusion algorithm developed by Shaw and Fox [14] and used by Lee [10]. In our version we take the normalized scores, but then further enhance scores for components appearing in both lists (doubling them) and penalize normalized scores appearing low in a single result list, while using the unmodified normalized score for higher ranking items in a single list.

2.5 Recalculation of LR Coefficients for Component Indexes

Using LR coefficients derived from relevance analysis of TREC data for INEX is unlikely to provide the most effective performance given the differences in tasks, queries and their structure, and relevance scales.

In order to begin to remedy this we have re-estimated the coefficients of the Logistic regression algorithm based on the INEX 2003 relevance assessments. In fact, separate formulae were derived for each of the major components of the INEX XML document structure, providing a different formula for each index/component of the collection. These formulae were used in only one of the official *ad hoc* runs submitted for the INEX 2004 evaluation, in order to have a basis of comparison with the fusion methods used in INEX 2002 and 2003. In this section we focus on the re-estimation and the values obtained for the new coefficients. Later we will this discuss the effectiveness of the new coefficients (or rather, the *lack* of effectiveness when used without supplementary adjustments) and several possible reasons for it.

For re-estimation purposes we submitted the INEX 2003 CO queries using the "Base" LR algorithm, which was the best performing LR-only experiment as reported in [9] (which was able to obtain 0.0834 *mean average precision* under the strict quantization, and 0.0860 under the generalized quantization). In addition we performed separate runs using only searches on single indexes (which may combine multiple document elements, as described in Tables 2 and 4). For all of these runs we captured the values calculated for each of the variables described in equation 4 for each document element retrieved. Then the *strict* relevance/non-relevance of each of these documents was obtained from the INEX 2003 relevance judgements and the resulting relevance/element data was analyzed using the SPSS logistic regression procedure to obtain re-estimations of the variable coefficients (b_i) in equation 4. The resulting coefficients for the various components/indexes are shown in Table 1, where the "Base" row is the default TREC-estimated coefficients and the other rows are the estimates for the named index. Not all indexes were reestimated because they (e.g., pauthor) tend to be used as purely Boolean criteria, or were components of another index and/or not present in all articles (e.g., kwd).

Testing these new coefficients with the INEX 2003 queries and relevance judgements we were able to obtain a mean average precision of 0.1158 under the strict metric and 0.1116 for the generalized metric, thus exceeding the best fusion results reported in [9]. However, the data used for training the LR model was obtained using the relevance data associated with the same topics, and it appears very likely that the model may be *over-trained* for that data, or that a different set of variables needs to be considered for XML retrieval.

3 INEX 2004 Adhoc Approach

Our approach for the INEX 2004 adhoc task was quite similar to that used for INEX 2003 runs. This section will describe the indexing process and indexes used, and also discuss the scripts used for search processing. The basic database was unchanged from last year's. We will summarize the indexing process and the indexes used in the adhoc task for reference in the discussion.

3.1 Indexing the INEX Database

All indexing in the Cheshire II system is controlled by an SGML Configuration file which describes the database to be created. This configuration file is subsequently used in search processing to control the mapping of search command index names (or Z39.50 numeric attributes representing particular types of bibliographic data) to the physical index files used and also to associated component indexes with particular components and documents. This configuration file also includes the index-specific definitions for the Logistic Regression coefficients (when not defined, these default to the "Base" coefficients shown in Table 1).

Table 2 lists the document-level (/article) indexes created for the INEX database and the document elements from which the contents of those indexes were extracted. These indexes (with the addition of proximity information the are the same as those used last year. The *abstract, alltitles, keywords, title, topic* and *topicshort* indexes support proximity indexes (i.e., term location), supporting phrase searching.

As noted above the Cheshire system permits parts of the document subtree to be treated as separate documents with their own separate indexes. Tables 3 & 4 describe the XML components created for INEX and the component-level indexes that were created for them.

Table 3 shows the components and the path used to define them. The component called COMP_SECTION consists of each identified section or subsection (<sec> ... </sec> or <ss*>... </ss*>) in all of the documents, permitting each individual section of an article to be retrieved separately. Similarly, each of the COMP_BIB, COMP_PARAS, and COMP_FIG components, respectively, treat

Table 1. Re-Estimated Coefficients for The Logistic Regression Model

Index	b_0	b_1	b_2	b_3	b_4	b_5	b_6
Base	-3.700	1.269	-0.310	0.679	-0.021	0.223	4.010
topic	-7.758	5.670	-3.427	1.787	-0.030	1.952	5.880
topicshort	-6.364	2.739	-1.443	1.228	-0.020	1.280	3.837
abstract	-5.892	2.318	-1.364	0.860	-0.013	1.052	3.600
alltitles	-5.243	2.319	-1.361	1.415	-0.037	1.180	3.696
sec_words	-6.392	2.125	-1.648	1.106	-0.075	1.174	3.632
para_words	-8.632	1.258	-1.654	1.485	-0.084	1.143	4.004

Table 2. Cheshire Article-Level Indexes for INEX

Name	Description	Contents
docno	Digital Object ID	//doi
pauthor	Author Names	//fm/au/snm, //fm/au/fnm
title	Article Title	//fm/tig/atl
topic	Content Words	//fm/tig/atl, //abs, //bdy, //bibl/bb/atl, //app
topicshort	Content Words 2	//fm/tig/atl, //abs, //kwd, //st
date	Date of Publication	//hdr2/yr
journal	Journal Title	//hdr1/ti
kwd	Article Keywords	//kwd
abstract	Article Abstract	//abs
author_seq	Author Seq.	//fm/au@sequence
bib_author_fnm	Bib Author Forename	//bb/au/fnm
bib_author_snm	Bib Author Surname	//bb/au/snm
fig	Figure Contents	//fig
ack	Acknowledgements	//ack
alltitles	All Title Elements	//atl, //st
affil	Author Affiliations	//fm/aff
fno	IEEE Article ID	//fno

Table 3. Cheshire Components for INEX

Name	Description	Contents
COMP_SECTION	Sections	//sec|//ss1|//ss2|//ss3|//ss4
COMP_BIB	Bib Entries	//bib/bibl/bb
COMP_PARAS	Paragraphs	//ilrj|//ip1|//ip2|, //ip3|//ip4|//ip5| //item-none|//p|//p1|//p2|//p3|fi//tmath|//tf
COMP_FIG	Figures	//fig
COMP_VITAE	Vitae	//vt

each bibliographic reference (<bb> ... </bb>), paragraph (with all of the alternative paragraph elements shown in Table 3), and figures (<fig> ... </fig>) as individual documents that can be retrieved separately from the entire document.

Table 4 describes the XML component indexes created for the components described in Table 3. These indexes make individual sections (COMP_SECTION) of the INEX documents retrievable by their titles, or by any terms occurring in the section. These are also proximity indexes, so phrase searching is supported within the indexes. Bibliographic references in the articles (COMP_BIB) are made accessible by the author names, titles, and publication date of the individual bibliographic entry, with proximity searching supported for bibliography titles. Individual paragraphs (COMP_PARAS) are searchable by any of the terms in the paragraph, also with proximity searching. Individual figures (COMP_FIG) are indexed by their captions, and vitae (COMP_VITAE) are indexed by keywords within the text, with proximity support.

Table 4. Cheshire Component Indexes for INEX †Includes all subelements of paragraph elements

Component or Name	Description	Contents
COMP_SECTION		
sec_title	Section Title	//sec/st
sec_words	Section Words	//sec
COMP_BIB		
bib_author	Bib. Author	//au
bib_title	Bib. Title	//atl
bib_date	Bib. Date	//pdt/yr
COMP_PARAS		
para_words	Paragraph Words	*†
COMP_FIG		
fig_caption	Figure Caption	//fgc
COMP_VITAE		
vitae_words	Words from Vitae	//vt

Almost all of these indexes and components were used during Berkeley's search evaluation runs of the 2004 INEX topics. The official submitted runs and scripts used in INEX are described in the next section.

3.2 INEX '04 Official Adhoc Runs

Berkeley submitted 5 retrieval runs for the INEX 2004 adhoc task, three CO runs and 2 VCAS runs. This section describes the individual runs and general approach taken in creating the queries submitted against the INEX database and the scripts used to do the submission. The paragraphs below briefly describe Berkeley's INEX 2004 runs.

Berkeley_CO_FUS_T_CMBZ (FUSION): This run uses automatic query generation with both Okapi BM-25 and Logistic regression retrieval algorithms combined using a score-normalized merging algorithm (MERGE_CMBZ). Results from multiple components where combined using MERGE_CMBZ as well. Separate retrieval of Articles, Sections and paragraphs were combined using score normalized merges of these results. Only Titles were used in generating the queries, which also included Boolean operations for proximity searching and "negated" terms. This run was based on the most effective fusion method found in our post-INEX 2003 analysis and reported in [9] and was intended as a baseline for comparison with the other runs.

Berkeley_CO_FUS_T_CMBZ_FDBK (FEEDBACK): This run is fundamentally the same as the previous run, with the addition of "blind feedback" where the <kwd> elements from top 100 results were extracted and the top 30 most frequently occurring keyword phrases were used as an addition to the base query generated by the initial query.

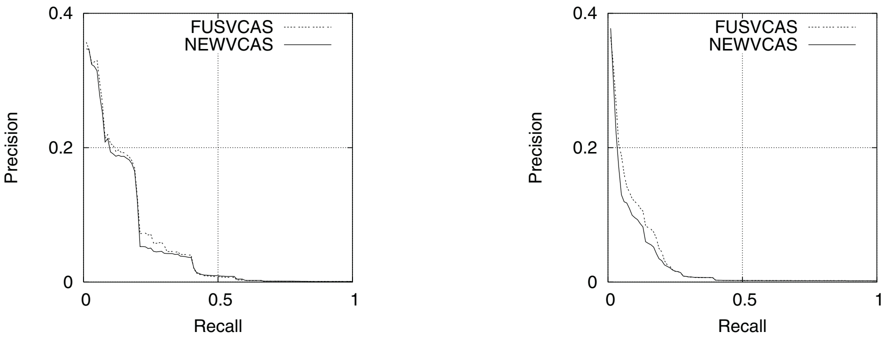


Fig. 1. Berkeley VCAS Runs – Strict (left) and Generalized (right) Quantization

Berkeley_CO_PROB_T_NEWPARMS (NEWPARMS): This run used automatic query generation with *only* the Logistic regression retrieval algorithm where the new coefficients for each of the indexes, as noted in Table 1, were used.

Berkeley_VCAS_FUS_T_CMBZ (FUSVCAS): This was a VCAS automatic run using automatic query generation from the NEXI title expression, and like the Berkeley_CO_FUS_T_CMBZ run, uses both Logistic Regression and the Okapi BM-25 ranking. Results from multiple components where combined using MERGE_CMBZ merging of results.

Berkeley_VCAS_PROB_T_NEWPARMS (NEWVCAS): This run also uses automatic query generation and is very similar to the NEWPARMS CO run above. Results from multiple components in this VCAS run were combined using the MERGE_CMBZ merger operator, as in the NEWPARMS CO run. This run used *only* the LR algorithm with the new LR coefficients as shown in Table 1.

Query Generation and Contents. All of the Cheshire client programs are scriptable using Tcl or Python. For the INEX test runs we created scripts in the Tcl language that, in general, implemented the same basic sequence of operations as described in the INEX 2003 paper[7]. For VCAS-type queries, the NEXI specification was used to choose the indexes (and components) to be searched, and the RESTRICT operators described above were used to validate proper nesting of components. For each specified "about" clause in the XPath, a merger of phase, keyword, Boolean and ranked retrieval was performed, depending on the specifications of the NEXI query.

3.3 INEX '04 Heterogeneous Track Runs

The Hetergeneous Track for INEX 2004 is attempting to test the ability to perform searches across multiple XML collections with different structures and contents. The evaluation results are still pending, so they cannot be discussed here. In this section we briefly describe the approach taken for the track and the system features used in the implementation.

Our approach to the Heterogeneous Track was to treat the different collections as separate databases with their own DTDs (simple "flat" DTDs were generated for those collections lacking them). The runs relied on Cheshire's "Virtual Database" features, in which multiple physical databases can be treated as if they were a single database. In addition we used the search attribute mapping features of the Z39.50 protocol, so that each physical database configuration file could specify that some subset of tags was to be used for "author" searches, another for "title", etc., for each as many of the index types described in Tables 2 and 4. Thus, when an "author" search was submitted to the virtual database, the query was forwarded to each of the physical databases, processed, and the results returned in a standardized XML "wrapper". Thus we were able to run scripts similar to those used for the adhoc track "CO" runs against the virtual database requesting the LR algorithm and obtain a result from all of the physical databases sorted by their estimated probability of relevance. In effect, the "Virtual Search" implements a form of distributed search using the Z39.50 protocol.

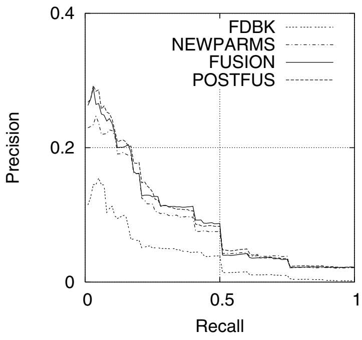
The only difficulty in this implementation was that all collections consisted of a single XML "document", including one of the databases where that single document was 217Mb in size. We ended up treating each of the main sub-elements of these "collection documents" as separate documents (another feature of Cheshire). The difficulty was then generating the actual full XPath for the elements in order to report results. This was eventually handled by a script that, in most cases, was able to infer the element from the internal document ID, and in the case of the 217Mb document (with multiple different subelements for the collection document) this involved matching each of the subtypes in separate databases. Until the evaluation is complete, we won't know whether this mapping was actually accurate.

4 Evaluation of Adhoc Submissions

The summary average precision results for the runs described above are shown in Table 5. The table includes an additional row (...POST_FUS_NEWPARMS) for an unofficial run that essentially used the Berkeley_CO_FUS_T_CMBZ structure

Table 5. Mean Average Precision for Berkeley INEX 2004

Run Name	Short name	Avg Prec (strict)	Avg Prec (gen.)
...._CO_FUS_T_CMBZ	FUSION	0.0923	0.0642
...._CO_FUS_T_CMBZ_FDBK	FEEDBACK	0.0390	0.0415
...._CO_PROB_T_NEWPARMS	NEWPARMS	0.0853	0.0582
...._VCAS_T_CMBZ	FUSVCAS	0.0601	0.0321
...._VCAS_PROB_T_NEWPARMS	NEWVCAS	0.0569	0.0270
...POST_FUS_NEWPARMS	POSTFUS	0.0952	0.0690



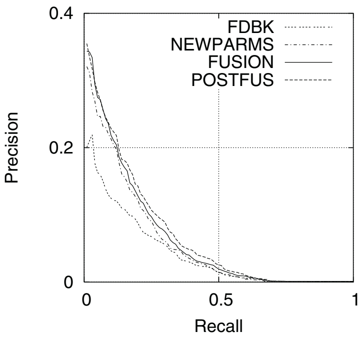


Fig. 2. Berkeley CO Runs – Strict and Generalized Quantization

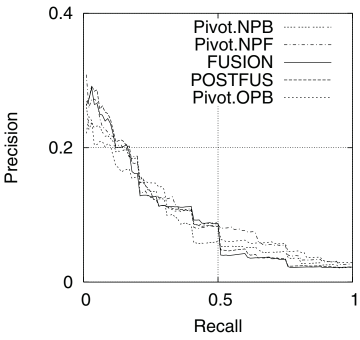
of combining LR and Okapi searching along with the new LR coefficients. This combination performed a bit better than any of the official runs. (However, see the next section about subsequent additional tests using pivoted component probabilities of relevance).

Figure 1 shows, respectively, the Recall/Precision curves for strict and generalized quantization of each of the officially submitted Berkeley "VCAS" runs. Figure 2 shows, respectively, the Recall/Precision curves for strict and generalized quantization of each of the officially submitted Berkeley "CO" runs. No Berkeley runs appeared in the top ten for all submitted runs. None of these runs was in the top 10, though the "FUSION" run was close (ranked 14th in aggregate score).

Our attempt at "blind feedback" performed very poorly (which was expected, given that it was very much a last-minute attempt, and we had no time to attempt to determine the optimal number of records to analyze or the number of retrieved <kwd> phrases to include in the reformulated query). More interesting was the fact that the re-estimated LR parameters, when used alone did not perform as well as the basic fusion method. However, when combined with in a fusion approach the new coeffients do improve the results over the basic Fusion method using the "Base" coefficients.

4.1 Additional Tests and Evaluation

Work reported at the INEX 2004 meeting by Mass and Mandelbrot[11] of the IBM Haifa Research Lab involving "pivoting" component weights by scaling them against the document-level weights for the same query appeared to offer a significant improvement in performance for the vector-space based algorithm used there. A similar pivot approach was reported by Sigurbjornsson, Kamps, and Rijke in the INEX 2003 Workshop Proceedings[15] in conjunction with a language model approach. We decided to apply the pivoted normalization method described in Mass and Mandelbrod[11] to the estimated probabilities for each component adjusted by the document-level probabilities returned by the new LR model parameters. We scale the estimated component probability of relevance based on a "Pivot" parameter, and the estimated probability of relevance for



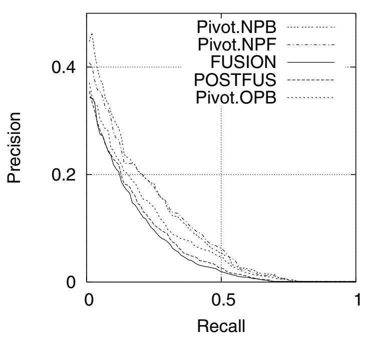


Fig. 3. – Strict (left) and Generalized (right) Quantization for Post-Official Tests

Table 6. Mean Average Precision for Additional Tests

Short name	Avg Prec (strict)	Avg Prec (gen.)	Avg Prec (aggr.)
PivotNPB	0.0901	0.1028	0.1146
PivotNPF	0.0961	0.0888	0.1108
PivotOPB	0.0849	0.0771	0.0944
POSTFUS	0.0952	0.0690	0.0819
FUSION	0.0923	0.0642	0.0776

the same search performed against the entire document. Formally, we estimate a new probability of relevance for the component, $P(R \mid Q, C')$, such that:

$$P(R \mid Q, C') = Pivot \cdot P(R \mid Q, D) + (1.0 - Pivot) \cdot P(R \mid Q, C) \qquad (6)$$

where $P(R \mid Q, C)$ is the original estimated probability of relevance for the component and $P(R \mid Q, D)$ is the estimated probability of relevance for the full document.

Table 6 shows the results of testing several variations of this "pivoted probabilities" approach. In all cases the article-level probability of relevance used for $P(R \mid Q, D)$ in equation 6 was taken from a search using the "topic" index described in Table 2. In all of the pivoted tests, the pivot value used was 0.70, the same value suggested by Mass and Mandelbrod in [11].

In Table 6 "PivotNPB" is an LR run using the new LR coefficients described above, with only the full text indexes of the section and paragraph component types described in Tables 3 and 3, along with the <article> and <bdy> tags. (That is, the same elements as used for the official runs, but not combining the results from multiple indexes on the elements.) The "PivotNPF" test is similar to the PivotNPB test, employing the new LR coefficients, but here the multiple index combinations used in the official runs were used for each component. The "PivotOPB" test is also similar to the PivotNPB test, except the old LR coefficients were used for all components and for the article-level probabilities

$P(R \mid Q, D)$. The "POSTFUS" and "FUSION" tests (the best performing tests from Table 5) were described in the previous section and are included in Table 6 for reference and comparison.

As can be seen in Table 6 the pivoted probabilities provide a considerable increase in the generalized performance metric over the FUSION and POSTFUS tests. The table also includes the "aggregate metric" merging all of the INEX 2004 metrics. The PivotNPB test provided the best generalized and aggregate metric performance, and the best performing test for the strict metric was the PivotNPF. Thus, with the addition of the pivoted element scores, the new LR coefficients do provide a noticeable increase in performance over the old coefficients. This is recent work and we not yet performed significance tests on the results, nor have we experimented with different settings for the pivot value.

5 Conclusions and Future Directions

We still need to perform a number of analyses of alternative combinations, but it appears that the re-estimated LR coefficients, although not as effective as the submitted FUSION approach when LR alone is used (without additional enhancement), do provide additional improvement when combined in a similar fusion approach, and when used alone with the addition of a document pivot as described in the previous section. However this improvement comes at the cost of a large increase in overlap between the returned elements. For example, the overlap for the FUSION test was 57.0640, while the overlap for the PivotNPB test was 92.03, suggesting that the primary effect of the pivot method is to promote elements with strong document matches in the ranking.

Our preliminary attempt at using blind feedback, with only assigned keywords in articles, doesn't appear to offer a benefit, though we plan to experiment a bit further using the framework developed for the "relevance feedback" track. Now with two years of usable and comparable relevance evaluations for INEX we can once again re-estimate the LR parameters from the INEX 2003 results and now test on the 2004, and also examine variations on the document pivot to attempt to determine optimal values for weighing elements vis a vis documents.

In addition we want to experiment with applying the score pivot approach to the Okapi algorithm and to new fusion combinations of LR and Okapi, combined with pivoted score normalization.

References

1. S. M. Beitzel, E. C. Jensen, A. Chowdhury, O. Frieder, D. Grossman, and N. Goharian. Disproving the fusion hypothesis: An analysis of data fusion via effective information retrieval strategies. In *Proceedings of the 2003 SAC Conference*, pages 1–5, 2003.
2. N. Belkin, P. B. Kantor, E. A. Fox, and J. A. Shaw. Combining the evidence of multiple query representations for information retrieval. *Information Processing and Management*, 31(3):431–448, 1995.

3. W. S. Cooper, F. C. Gey, and A. Chen. Full text retrieval based on a probabilistic equation with coefficients fitted by logistic regression. In D. K. Harman, editor, *The Second Text Retrieval Conference (TREC-2) (NIST Special Publication 500-215)*, pages 57–66, Gaithersburg, MD, 1994. National Institute of Standards and Technology.
4. W. S. Cooper, F. C. Gey, and D. P. Dabney. Probabilistic retrieval based on staged logistic regression. In *15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Copenhagen, Denmark, June 21-24*, pages 198–210, New York, 1992. ACM.
5. R. R. Larson. TREC interactive with cheshire II. *Information Processing and Management*, 37:485–505, 2001.
6. R. R. Larson. A logistic regression approach to distributed IR. In *SIGIR 2002: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 11-15, 2002, Tampere, Finland*, pages 399–400. ACM, 2002.
7. R. R. Larson. Cheshire II at INEX: Using a hybrid logistic regression and boolean model for XML retrieval. In *Proceedings of the First Annual Workshop of the Initiative for the Evaluation of XML retrieval (INEX)*, pages 18–25. DELOS workshop series, 2003.
8. R. R. Larson. Cheshire II at INEX 03: Component and algorithm fusion for XML retrieval. In *INEX 2003 Workshop Proceedings*, pages 38–45. University of Duisburg, 2004.
9. R. R. Larson. A fusion approach to XML structured document retrieval. *Journal of Information Retrieval*, 2005. (in press).
10. J. H. Lee. Analyses of multiple evidence combination. In *SIGIR '97: Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, July 27-31, 1997, Philadelphia*, pages 267–276. ACM, 1997.
11. Y. Mass and M. Mandelbrod. Component ranking and automatic query refinement for xml retrieval. In *INEX 2004 Workshop Pre-Proceedings*, pages 134–140. University of Duisburg, 2004.
12. S. E. Robertson and S. Walker. On relevance weights with little relevance information. In *Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 16–24. ACM Press, 1997.
13. S. E. Robertson, S. Walker, and M. M. Hancock-Beauliee. OKAPI at TREC-7: ad hoc, filtering, vlc and interactive track. In *Text Retrieval Conference (TREC-7), Nov. 9-1 1998 (Notebook)*, pages 152–164, 1998.
14. J. A. Shaw and E. A. Fox. Combination of multiple searches. In *Proceedings of the 2nd Text REtrieval Conference (TREC-2), National Institute of Standards and Technology Special Publication 500-215*, pages 243–252, 1994.
15. B. Sigurbjrnsson, J. Kamps, and M. de Rijke. An element-based approach to xml retrieval. In *INEX 2003 Workshop Proceedings*, pages 19–26. University of Duisburg, 2004.