Published and presented at IEEE – ITTC 05 – Las Vegas – Abril 205

# Web Services for Information Retrieval

João Ferreira
*Instituto Superior de Engenharia de Lisboa*
*jferreira@deetc.isel.ipl.pt*

Alberto Rodrigues da Silva
*INESC-ID, Instituto Superior Técnico*
*silva@acm.org*

José Delgado
*Instituto Superior Técnico*
*delgado@tagus.ist.utl.pt*

## Abstract

*The idea developed in this paper is the creation of standard information retrieval modules in a distributed manner in order to create testing systems to validate methods and algorithms. An investigator in information retrieval can construct a retrieval test system just by integrating different modules and manipulating the input variables of each module.*

## 1. Introduction

How do we find information on the Web? This is an old issue far from being solved. Several methods and algorithms have been tested without complete satisfactory results [1, 2]. Each time a new method or algorithm is created, a big effort is applied in the construction of a retrieval system to test that method or algorithm. There are no standard retrieval systems and just a few open sources that can support different retrieval models in a modular way. In figure 1, we list the most relevant open source retrieval systems and their scope of utility (regarding retrieval models). All these systems are oriented towards one specific retrieval module developed by a specific investigation group. A system that has more than one retrieval module (e.g. Lemur, Terrier) uses additional modules for comparison of results. This paper proposes a standard modular platform that can be configured to different retrieval, filtering and classification systems. This work can be the starting point of a standard retrieval system that can be available in a distributed manner and controlled by organisms like TREC. The main advantages of standard modular retrieval systems are: (1) less work for testing methods and algorithms; (2) the common infrastructure provides more reliable comparison of results; (3) the distribution ensures the mobility and the possibility of having better collaborative work in the information retrieval research field. Our modular platform was constructed to support different retrieval models (e.g. vectorial, probabilistic, link analysis and classification) and to explore the fusion of results from different models. It is different from others because it covers more IR models and it is focussed on the development of standards.

| System | Year | Written | Developed | IR Models Implemented |
|---|---|---|---|---|
| Smart | 1960 | C | Cornell | Vector Space with several weighthing options |
| Terrier | 2001 | Java+ perl | Glasgow | Probabilistic framework (DFD); Hyperlinks; combinations; query expansion |
| Okapi | 1992 | C | City Unv. London (CISR) | Probabilistic |
| Lemur | 2000 | C;C++ | CMU and Univ. Massachusetts | Space \ Probabilistic models includes Language modeling |
| Inquery | 1994 | C | Massachusets | Bayesian inference |
| Zettair (lucy) | | C | RMIT | Boolean, ranked and phrase querying |
| MG | 1990 | C | U. Waikato, U. Melbourne and RMIT | Vector Space with compression and speed up tricks |

**Figure 1: Most relevant open source retrieval systems.**

## 2. Retrieval Process

In this paper, an IR module is considered as a starting point and once accepted as a standard can become a service. The main modules of an IR framework are:

- indexing, responsible for reducing and storing the information space;
- matching, which compares information space representatives with user information needs;
- link analysis from a set of relevant document follow links and based on hub and authorities measures identifies a new set of relevant documents;
- classification, where documents are catalogued on a classified system (knowledge space of a certain domain, previously defined) and information needs performed on this organized space;

- feedback, to expand users' queries which are usually reduced.
- fusion of different methods and systems parameters through fusion formulas.

The Communication Space follows the standards of W3C for WebServices and allows communications between the different modules to exchange information based on this standard. This distributed environment should allow investigators to share modules and results and also to constructed systems in a collaborative way.

### 2.1. Indexing

- The main modules of the indexing process are:
- Text process. Our IR framework processes only text, so all formats should be converted. Pdf and word formats in our case were converted by pdf2text, html2text, but other choices were valid;
- Filters: identify specific information like headers (frequency of these terms are multiplied by 10), full document, partial document (defined size), phrases and URL URL, phrases based on electronic dictionary. One possible solution that we applied was the dictionary of Roger Mitton, Oxford Advanced Learner to identify phrases <www.oup.com/elt/global/products/oald/>
- Stop words are removed based on the list of the 390 not relevant terms released by WAIS (*Wide Area Information System*) <www.ai.mit.edu/extra/thenet/wais.html>
- Snowball <snowball.tartarus.org>: stemming program in which we implemented Porter's algorithm. Snowball provides different language stemmers.
- Weight calculation: from term frequency. Several approaches can be implemented.

The Indexing module was created based on: (1) OpenFTS <openfts.sourceforge.net>, front-end which integrates snowball, stop words removal, phrase construction through dictionary and user interface; (2) tsearch2 <www.sai.msu.su/~megera/postgres/gisttsearch/V2> have pre-defined filters, interface to data base through tsearch vector; (3) weight term calculation: several approaches based on term frequency are easily implemented; (4) data base postgresql. The modules in white must be built. The grey modules are common and just need to be integrated.

The user (IR investigator) has the following variables to control and choices available:

- Indexed information from: (1) entire document (2) the document body (can define the size or the maximum number of indexed terms per document); (3) the document header; (4) use phrase (yes or no) and if yes choose number of word proximity to form the phrase (5) URL.
- Choose stemming algorithm and language
- Choose stop words and stop URL file
- Choose the dictionary to be integrated
- Choose weight calculation scheme

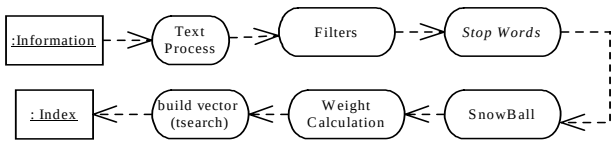


**Figure 2: Description of indexing process**

### 2.2. Matching

Uses information stored as representative of information and compares it with the user needs expressed in the query. The main inputs to these modules are: (1) Matching formula, that can be vectorial (with cosine measure), Boolean; (2) Relevance measurement, which is the measure from each document that is considered relevant.

$$\mathbf{q}^{\mathrm{T}}\mathbf{d}_i = \sum_{k=1}^{t} q_k d_{ik} \quad , \qquad (1)$$

where $q_k$ is the weigh of the term $k$ on query, $d_{ik}$ is the weight of term $k$ on documents $i$ (i.e. *Lnu*) and $t$ is the number of terms in the collection. The Output is stored back in the database. Again, different matching algorithms can be easily implemented.

### 2.3. Feedback

This module represents a complete approach for this subject. As input, we can have user judgments and documents relevant identified by system. Mainly, this module adds new terms or changes the weights of terms from the query. We implemented an approach in which the top ten positive and top two negative weighted terms, from the top three ranked documents of the initial retrieval results, were used to expand the initial query in a pseudo-feedback retrieval process based on the adaptive linear model. The basic approach of the adaptive linear model, which is based on the concept of preference relations from decision theory [3], is to find a *solution vector* that will rank a more-preferred document before a less-preferred one (Wong et al., 1988).

The solution vector is obtained via an *error-correction procedure*, which begins with a *starting vector* $\mathbf{q}_{(0)}$ and repeats the cycle of "error-correction" until a vector that ranks documents according to the preference order estimation based on relevance feedback is found [4]. The error-correction cycle $i$ is defined by: $\mathbf{q}_{(i+1)} = \mathbf{q}_{(i)} + \alpha\mathbf{b}$ , (2); where $\alpha$ is a constant, and $\mathbf{b}$ is the *difference vector* resulting from subtracting a less-preferred document vector from a more preferred one [5]. The choices for the constant $\alpha$ and the *starting vector* $\mathbf{q}_{(0)}$ are taken from various experiences on TREC [5, 6]. Again other models of feedback can be easily implemented.

### 2.4. Link analysis

The connections model identifies in the first step the seed groups using the vectorial model. The hub and authority measurements of these groups are calculated through the HITS (Hyperlink Induced Text Search) algorithm [7].

The address definition was based on the document's URL cutting it in the first occurrence of the mark ("/") (short address format) and the long format until the last occurrence of the mark ("/").

The main system parameters are the choice of the method to create the seed group, the address definition and the way hubs and authorities are measured. The total number of possible systems is 2x3=6.

$$a(p) = \sum_{q \to p} h(q) \times auth\_wt(q,p), \quad (3)$$

$$h(p) = \sum_{p \to q} a(q) \times hub\_wt(p,q), \quad (4)$$

In these equations, *auth_wt(q,p)* is 1/*m* for page *q*, whose host has *m* documents pointing to *p*, and *hub_wt(p,q)* is 1/*n* for page *q*, which is pointed by *n* documents from the host of *p*.

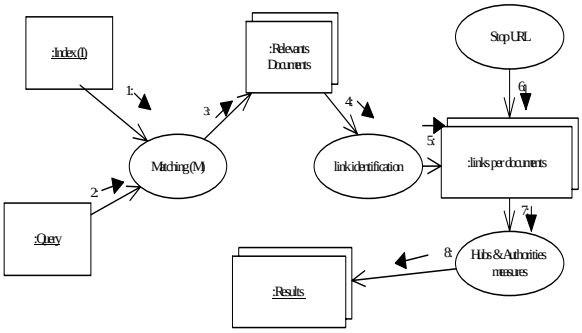


**Figure 3: Main modules of link analyses system**

### 2.5. Probabilistic approach

The probabilistic approach involves a two-step process of contingency table construction and association weight calculation. If each document $\boldsymbol{D_i}$ in a collection is regarded as a multi-set $a_i$ of $m$ document terms and $b_j$ of $n$ query terms, i.e. $\boldsymbol{D}_i = (\{a_{i1},\ldots,a_{im}\};\{b_{j1},\ldots,b_{jn}\})$, the associations contained in a particular document $\boldsymbol{D}_i$ consist of all the ordered pairs that can be formed from $a_{im}$X $b_{jn}$ document subparts. For each term $A$ and a query $B$ (i.e. $a_{im}$-$b_{jn}$ pair), a contingency table is formed containing the counts for each of the possible combinations of $A$ and $B$:

| | |
|---|---|
| AB | A $\bar{B}$ |
| $\bar{A}$ B | $\bar{A}$ $\bar{B}$ |

where "-" denotes the absence of some event. As each of the pairs for each document is considered, contingency tables are constructed and updated. When all the pairs and contingency tables have been recorded after processing all the documents in a collection, the strength of the associations can be computed for each document_term-query_term pair using a likelihood ratio statistic as the measure of association. The strength of association is computed by the following formula [8]:

$$\lambda = 2\left[\ln\frac{L(p_1,k_1,n_1)}{L(p,k_1,n_1)} + \ln\frac{L(p_2,k_2,n_2)}{L(p,k_2,n_2)}\right], \quad (5)$$

where: $\ln\left(L(p,k,n)\right) = k\ln p + (n-k)\ln(1-p)$;

$$p_1 = \frac{k_1}{n_1};\ p_2 = \frac{k_2}{n_2};\ p = \frac{k_1+k_2}{n_1+n_2};\ k_1 = AB,$$

$n_1 = AB + \bar{A}\,B,\ k_2 = A\,\bar{B}$ , *and* $n_2 = A\,\bar{B} + \bar{A}\ \bar{B}$ .

For each entry document_term-query_term pair, the strength of association is calculated. Documents are ordered by the sum of all document terms associated with query terms like: document_$term_1$-query_$term_1$ with weights $w_1$ and document_$term_1$-query_$term_2$ with weights $w_2$ can be collapsed into document_$term_1$ with weights $(w_1 + w_2)$.

### 2.6. Classification

The Web Directory search was implemented based on the Term Match (TM) method. TM takes a simpler approach of finding categories in which query terms occur by extending the typical category search

implementation of Web directory services or specific classified systems.

The first phase produces a ranked list of categories for a query, matches query terms to terms in the classification system to find a set of matching nodes in the classification hierarchy and generates a ranked category list in the following manner:

1. For each matching category, (i) compute *#tfc* (number of unique query terms in the category label); (ii) compute *#tfs* (number of unique query terms in the site title and description) in all its sites; (iii) compute *psc* (proportion of sites with query terms in the category).

2. Rank the matching categories in the descending order of *#tfc*, *#tfs*, and *psc*.

Note that categories ranked via sorting by multiple variables in such an order that the terms in category labels, which are likely to be highly "powerful", are given precedence over terms in site titles or descriptions. This ranking approach is similar to how Yahoo ranks its search results except that it combines the category and site match results while collapsing the site match results to their parent categories.

The second phase of the TM method is to expand query vector (the class centroid in the TM method) that is built from the best matching categories to produce a ranked list of the collection of documents. The expanded query vector of the TM method is a vector of selected category terms with normalized term-category association weights. The parameters tested for the TM systems are the number of top categories used, the collection term index and terms for pseudo-feedback.

### 2.7. Fusion

The main inputs are fusion formulas and systems or internal systems parameters to combine.

We implemented two of the most common fusion formulas, the *Similarity Merge* [9, 10, 11, 12] and *Weighted Sum* [13, 14, 15, 16].

The Similarity Merge (SM) fusion formula, originally introduced by [9, 10] and refined by [11, 12], computes the fusion score of a document by the sum of normalized component scores boosted by the retrieval overlap. In order to address the issue of combining a large number of systems with uneven distribution across methods, the overlap count was normalized by the number of systems in a given method. Equation (6) below describes the SM fusion formula used to merge and rank documents retrieved by different systems: $FS = (\sum NS_i) * \frac{\#sist}{m(i)}$ (6); where:

*FS* = fusion score of a document;

$NS_i = (S_i - S_{min}) / (S_{max} - S_{min})$ (7);

$NS_i$ = normalized score of a document by system *i*, is computed by Lee's min-max formula [11,12], where $S_i$ is the retrieval score of a given document and $S_{max}$ and $S_{min}$ are the maximum and minimum document scores by system *I; #sist* = number of systems that retrieved a given document; *m*(*i*) = number of systems in a method to which system *i* belongs.

To compensate for the differences among fusion component systems, we tested the *Weighted Rank Sum* (WRS) formula, which uses rank-based scores (e.g. 1/rank) in place of document scores of WS formula:

$FS = \sum(w_i * RS_i)$, (6); where: *FS* = fusion score of a document, $w_i$ = weight of system *i*, $RS_i$ = rank-based score of a document by system *i*.

## 3. System

A retrieval system is constructed with the integration of the modules shown in figure 4. The investigator can easily choose variables and parameters to use in each module. Crawler is implemented through Larbin, <larbin.sourceforge.net>. User feedback is incorporated with the identification of target information sources. The communication module establishes the communications with the standards defined for Web Services at W3C <www.w3c.org/2002/ws>.

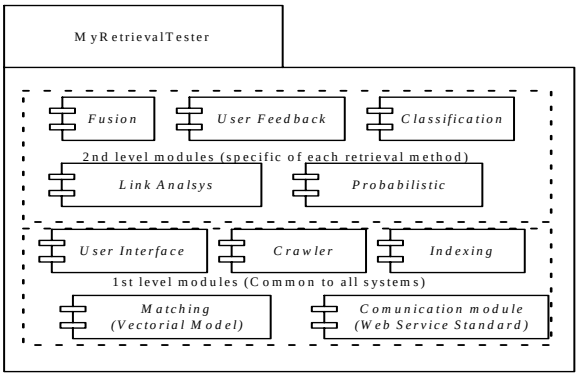


**Figure 4: Main modules of MyRetrievalTester system.**

With system evolution and use, several approaches to do the same task will be available and this kind of environment will support a better collaboration between information retrieval (IR) investigators. In the next section we will show experimental results produced easily from this platform, just by integrating modules and choosing terminal variables.

## 4. Results

To control and have measures of systems performance we use a controlled environment provided by TREC. As the source for these experiences we use the WT10g collection [17], which is a ten-gigabyte subset of the 1997 Web crawl by the Internet Archive, consists of 1.7 million Web documents, 50 TREC queries (topics 451-500), and official NIST relevance judgments. The WT10g collection also includes the connectivity data, which provides lists of inlinks and outlinks of all documents in the collection.

As classification systems for the Web lack an ideal Web directory, we use Yahoo <http://yahoo.com> due to its size and popularity.

The vectorial model uses the following parameters; (1) query length (long(l), medium(m) and small(s)), (2) indexing information from headers(t), body document (c) or complete document (d), (3) pharos (yes or no); (4) user feedback(1(yes)/0(no)). Combination of all parameters gives 36 different systems.

Link analysis with the following parameters: (1) host definition (l-long; s-small) (2) seed set (vectorial, (l-long; m-medium; s-small) query, body text, phrase, no feedback. Combination of all parameters gives 6 different systems.

Probabilistic approach with the following parameters: (1) query length (s/m/l); WT10g index (1 – header with phrase; 2-document with phrase; 3 – header without phrase; 4-document without phrase); feedback (1(yes)/0(no)). Combination of all parameters gives 12 different systems.

Classification with the following parameters: # top categories (1/2/3); WT10g index (body text, no phrase (*1*); body text, phrase (*2*); body+header, no phrase (*3*); body+header, phrase (*4*) ); feedback (1(yes)/0(no)). Combination of all parameters gives 24 different systems.

The recall-precision curves show (Figure 5) marked differences in performance across methods. In fact, the average precision values of top systems diminish roughly by half in each method order from VSM and probabilistic, TM, HITS, thus indicating the overpowering advantage of text-based method over other methods. The poor results of link analyses seem to indicate an incomplete link structure in the WT10g collection. For complete results, see [18].

Legend for figure 5:

- *vlc10* (vectorial with: long query, body text, phrase, and no feedback )
- *hsm* (HITS with: short host, seed set system of *vmc10*)
- *pl41* (Probabilistic with, long query document without phrase and with feedback;
- *t221* (top 2 categories, body text, phrase, no feedback)

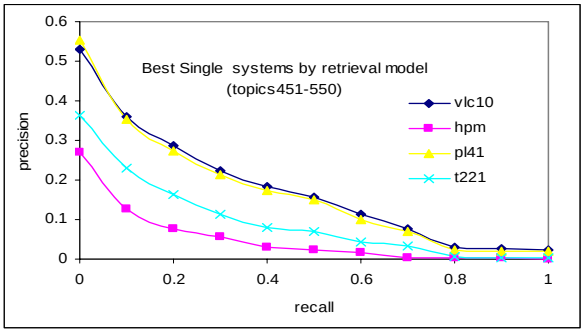


**Figure 5: Best results of each simple system for topics 451-550.**

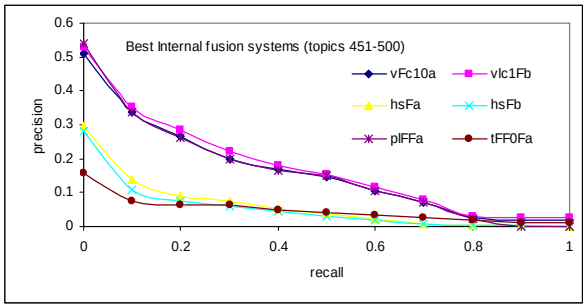


**Figure 6: Best internal fusion systems for topics 451-500.**

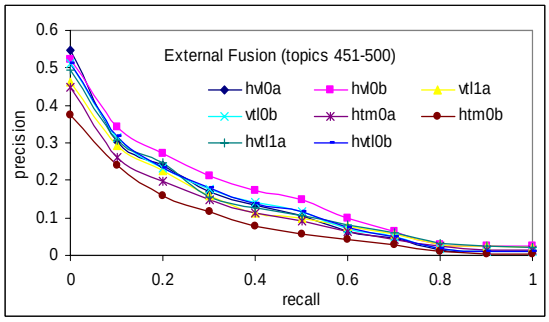


**Figure 7: External Fusion systems for topics 451-500.**

We have combined all internal parameters through SM and WRS formulas and additionaly combined external systems in 2- and 3-way cross-method (i.e. VSM-HITS, VSM-TM, HITS-TM, VSM-HITS-TM). Since Probabilistic systems had a similar behavior to that of VSM, it was dropped from the fusion pool in external fusion. The best result of each type of fusion results is represented in figure 6 (internal fusion) and figure 7 (external fusion). *F* means combination of all parameters, suffix *a* means fusion performed through

SM formulae and *b* means fusion by WRS formulae. On external fusion we use common parameters such as query and feedback.

Internal fusion produced results in between upper and lower threshold performance levels determined by the baseline systems of the methods combined. As in the case of the intra-method fusion, introduction of the TM system results into the fusion pool degraded the performance level of the combined results in all external fusion combinations, with the exception of the HITS-TM fusion, in which diverse solution spaces of HITS systems seemed to overpower the potential adverse influences of WD systems to produce the fusion results that surpassed the baseline performance level. The combination of VSM and HITS systems, however, did not produce better results than the baseline, because the solution space of HITS systems, though diverse from one another and from the solution spaces of TM systems, had much overlap with those of VSM systems.

The different results of SM and WRS fusion formulas, which were observed in internal fusion, appeared in external fusion results as well, although the SM formula results seemed to be more stable across methods than WRS formula results. In general, the WRS formula appeared to have an advantage over the SM formula, which worked better with HITS systems.

## 5. Conclusions

We demonstrated the possibility of building Web Services for information retrieval with a modular structure. The purpose of this system is to provide a standard and collaborative tool used in the investigation of methods and algorithms in IR field of activity. Systems can be easily created and changed. One investigator from IR can test different models and algorithms by integrating or changing some part of the platform. Also the platform simplifies the evaluation process since there are common sharable parts between the different system proposed. System´s distribution facilitates collaboration between different IR groups.

## 6. References

[1] Korfhage Robert R.. Information Storage and Retrieval. John Wiley e Sons Inc., 1997.

[2] Yates R. B. E Neto B. R.. Modern Information Retrieval**.** Addison-Wesley Pub Co., 1999.

[3] Fishburn P. C.. Utility theory for decision making, John Wiley, New York, 1970.

[4] Wong S. K. M. Yao Y. Y. Salton G. & Buckley C.. Evaluation of an adaptive linear model. JASIS, 1991, 42 723-730.

[5] Sumner, R. G., Jr., & Shaw, W. M., Jr.. An investigation of relevance feedback using adaptive linear and probabilistic models. In E. M. Voorhees & D. K. Harman (Eds.), *The Fifth Text REtrieval Conference (TREC-5)*, 1997.

[6] Sumner, R. G., Jr., Yang, K., Akers, R., & Shaw, W. M., Jr.. Interactive retrieval using IRIS: TREC-6 experiments. In E. M. Voorhees & D. K. Harman (Eds.), *The Sixth Text REtrieval Conference (TREC-6)*, 1998.

[7] Kleinberg, J.. Authoritative sources in a hyperlinked environment. *Proceeding of the 9th ACM-SIAM Symposium on Discrete Algorithms*, 1997.

[8] Sparck K.J.,. Walker S,. Robertson, S.E. A Probabilistic Model of Information Retrieval: Development and Status. Information Processing and Management, (1998).

[9] Fox E. A. & Shaw J. A. Combination of multiple searches. In D. K. Harman (Ed.). TREC-2, 1994.

[10] Fox, E. A., & Shaw, J. A. Combination of multiple searches. In D. K. Harman (Ed.), The Third Text Rerieval Conference (TREC-3) Washington, DC: U.S. Government Printing Office, 1995, NIST Spec. Publ. 500-225, 105-108.

[11] Lee J. H.. *Combining multiple evidence from different relevance feedback methods (Tech. Rep. No. IR-87).* Amherst: University of Massachusetts Center for Intelligent Information Retrieval, 1996.

[12] Lee J. H. Analyses of multiple evidence combination. Proceedings of the ACM SIGIR Conference on Research and Development in IR, 1997, 267-276.

[13] Bartell, B. T., Cottrell, G. W., & Belew, R. K.. Automatic combination of multiple ranked retrieval systems. Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval, 1994.

[14] Larkey, L. & Croft, W. B.. Combining Classifiers in Text Categorization. *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, 1996, 289-297.

[15] Modha, D. & Spangler, W. S.. Clustering hypertext with applications to Web searching. *Proceedings of the 11th ACM Hypertext Conference*, 2000,143-152.

[16] Thompson. P.. A combination of expert opinion approach to probabilistic information retrieval, part 1: The conceptual model. *Information Processing & Management*, 26(3), 1990, 371-382.

[17] www.ted.cmis.csiro.au/TRECWeb/ access_to_data.html

[18] Ferreira, J.. Retrieval Information on the Internet. PhD Thesis, IST, Portugal (in Portuguese), 2004.