University of Glasgow at TREC2004:
Experiments in Web, Robust and Terabyte tracks with Terrier

Vassilis Plachouras, Ben He, and Iadh Ounis

University of Glasgow, G12 8QQ Glasgow, UK

Abstract

With our participation in TREC2004, we test Terrier, a modular and scalable Information Retrieval framework, in three tracks. For the mixed query task of the Web track, we employ a decision mechanism for selecting appropriate retrieval approaches on a per-query basis. For the robust track, in order to cope with the poorly-performing queries, we use two pre-retrieval performance predictors and a weighting function recommender mechanism. We also test a new training approach for the automatic tuning of the term frequency normalisation parameters. In the Terabyte track, we employ a distributed version of Terrier and test the effectiveness of techniques, such as using the anchor text, pseudo query expansion and selecting different weighting models for each query.

1 Introduction

With our participation in TREC2004, we test our Information Retrieval (IR) framework, Terrier, in a variety of different settings. Terrier is a modular and scalable framework, for the rapid development of large-scale IR applications. It provides indexing and retrieval functionalities, as well as a number of parameter-free weighting models, based on the Divergence From Randomness (DFR) framework [2]. Terrier stands for TErabyte RetrIEveR, and further information can be found at http://ir.dcs.gla.ac.uk/terrier.

We have submitted official runs to three tracks of TREC2004. For the Web track, we test the selective application of different retrieval approaches on a per-query basis. In the Robust track, we employ two novel pre-retrieval performance predictors in a weighting function recommender mechanism, in order to use the optimal weighting functions/models for the poorly-performing queries. We also refine the automatic tuning of the term frequency normalisation parameters, by creating samples of queries, instead of using relevance information. In both the Web and Robust tracks, we use a centralised version of Terrier. For the Terabyte track, we use Terrier in a distributed setting, in order to handle the test collection .GOV2, and evaluate retrieval techniques, which have been effective in the context of previous adhoc and Web retrieval TREC tasks.

The remainder of the paper is organised as follows. Section 2 contains a description of the Terrier framework. In Section 3, we describe our approach for the mixed query task of the Web track. Section 4 presents our experiments for the Robust track. In Section 5, we describe our participation in the Terabyte track, and we close with some concluding remarks in Section 6.

2 Terrier Information Retrieval Framework

Terrier offers a number of DFR-based models, classical and recent models for document weighting, as well as DFR-based and classical term weighting models for query expansion. More specifically, the relevance score of a document d for a particular query Q is given by:

$$score(d, Q) = \sum_{t \in Q} w(t, d) \tag{1}$$

where $w(t, d)$ is the weight of the document d for a query term t. In Table 1, we provide the formulas for the different models $w(t, d)$ we have used in our experiments for TREC2004.

Model	Formula
BB2	$w(t,d) = \frac{F+1}{N_t \cdot (tfn+1)}\big(-\log_2(N-1) - \log_2(e) + f(N+F-1, N+F-tfn-2) - f(F, F-tfn)\big)$
BL2	$w(t,d) = \frac{1}{tfn+1}\big(-\log_2(N-1) - \log_2(e) + f(N+F-1, N+F-tfn-2) - f(F, F-tfn)\big)$
PB2	$w(t,d) = \frac{F+1}{N_t \cdot (tfn+1)}\big(tfn \cdot \log_2 \frac{tfn}{\lambda} + (\lambda + \frac{1}{12 \cdot tfn} - tfn) \cdot \log_2 e + 0.5 \cdot \log_2(2\pi \cdot tfn)\big)$
PL2	$w(t,d) = \frac{1}{tfn+1}\big(tfn \cdot \log_2 \frac{tfn}{\lambda} + (\lambda + \frac{1}{12 \cdot tfn} - tfn) \cdot \log_2 e + 0.5 \cdot \log_2(2\pi \cdot tfn)\big)$
I(n)B2	$w(t,d) = \frac{F+1}{N_t \cdot (tfn+1)}\big(tfn \cdot \log_2 \frac{N+1}{N_t+0.5}\big)$
I(n)L2	$w(t,d) = \frac{1}{tfn+1}\big(tfn \cdot \log_2 \frac{N+1}{N_t+0.5}\big)$
I(F)B2	$w(t,d) = \frac{F+1}{N_t \cdot (tfn+1)}\big(tfn \cdot \log_2 \frac{N+1}{F+0.5}\big)$
I(F)L2	$w(t,d) = \frac{1}{tfn+1}\big(tfn \cdot \log_2 \frac{N+1}{F+0.5}\big)$
I(n_e)B2	$w(t,d) = \frac{F+1}{N_t \cdot (tfn+1)}\big(tfn \cdot \log_2 \frac{N+1}{n_e+0.5}\big)$
I(n_e)L2	$w(t,d) = \frac{1}{tfn+1}\big(tfn \cdot \log_2 \frac{N+1}{n_e+0.5}\big)$
I(n_e)C2	$w(t,d) = \frac{F+1}{N_t \cdot (tfn_e+1)}\big(tfn_e \cdot \log_2 \frac{N+1}{n_e+0.5}\big)$

Table 1: Terrier DFR-based document weighting models

The notation from Table 1 is explained below:

- tf is the within-document frequency of term t in document d.
- F is the term frequency of term t in the whole collection.
- N is the number of documents in the collection.
- N_t is the document frequency of term t.
- n_e is given by $N \cdot \big(1 - (1 - \frac{N_t}{N})^F\big)$.
- λ is given by $\frac{F}{N}$ and $F \ll N$.
- The relation f is given by the Stirling formula:

$$f(n, m) = (m + 0.5) \cdot \log_2(\frac{n}{m}) + (n - m) \cdot \log_2 n \tag{2}$$

- tfn is the normalised term frequency. It is given by the *normalisation 2*:

$$tfn = tf \cdot \log_2(1 + c \cdot \frac{avg\_l}{l}) \tag{3}$$

where c is a parameter. l and avg_l are the document length of the document d and the average document length in the collection respectively.

- tfn_e is also the normalised term frequency. It is given by the modified version of the normalisation 2:

$$tfn_e = tf \cdot \log_e(1 + c \cdot \frac{avg\_l}{l}) \tag{4}$$

The only free parameter of the DFR framework is the term frequency normalisation parameter c from Eqs. 3 and 4. The tuning of such parameters is a crucial issue in information retrieval, because it has an important impact on the retrieval performance [5, 2]. A classical tuning method is the pivoted normalisation [16], which fits the document length distribution to the length distribution of relevant documents. However, since document length distribution is collection-dependent, the pivoted normalisation suffers from the collection-dependency problem. Indeed, the optimal parameter settings of diverse document collections are different [5].

In our experiments with Terrier, the parameter c is automatically tuned, according to a method proposed by He and Ounis [10]. This method assumes a constant optimal normalisation effect with respect to the document length distribution of the collection, and it assigns the parameter value such that it gives this constant. Thus, it is a collection-independent approach. The proposed method in [10] uses relevance information for training.

Terrier provides various DFR-based models for query expansion, based on extracting the most informative terms from a set of top-ranked documents. In Table 2, we present the term weighting models $w(t)$ used in our experiments for TREC2004.

Model	Formula
KL	$w(t) = P_x \cdot \log_2 \frac{P_x}{P_c}$
CS	$w(t) = l_x \cdot D + 0.5 \cdot \log_2(\pi \cdot l_x \cdot \frac{1-tf_x}{token_c})$
Bo1	$w(t) = tf_x \cdot \log_2 \frac{1+P_n}{P_n} + \log_2(1 + P_n)$
Bo2	$w(t) = tf_x \cdot \log_2 \frac{1+P_f}{P_f} + \log_2(1 + P_f)$

Table 2: Terrier DFR-based term weighting models.

The notation from Table 2, is explained below:

- l_x is the sum of the length of the exp_doc top-ranked documents, and exp_doc is a parameter of the query expansion methodology.
- tf_x is the frequency of the query term in the top-ranked documents.
- $token_c$ is the total number of tokens in the whole collection.
- P_n is given by $\frac{F}{N}$, where F is the term frequency of the query term in the whole collection and N is the number of documents in the whole collection.
- P_f is given by $\frac{tf_x \cdot l_x}{token_c}$.
- D is given by:

$$P_x \cdot \log_2 \frac{P_x}{P_c} + P_x \cdot \log_2 \frac{1 - P_x}{1 - P_c} \tag{5}$$

  where $P_x = tf_x/l_x$ and $P_c = \frac{F}{token_c}$.

3 Web Track

Our experiments for the Web track of TREC2004 continue the evaluation of a decision mechanism for the dynamic application of appropriate retrieval approaches on a per-query basis. We use Terrier, a modular Information Retrieval framework and its associated DFR-based weighting models, as described in Section 2.

We have submitted runs for the mixed query task of the Web track. In this task, there are 225 topics, which can be either topic distillation, named page finding, or homepage finding topics. The queries are created from the title of each topic. However, the system is not aware of the actual type of each query, during retrieval. This task is more similar to the operational setting of a Web search engine, which receives user queries without explicit evidence of the query type. Our aim is to use a decision mechanism for selecting an appropriate retrieval approach for each query, based on evidence from the hyperlink structure and the anchor text of the set of retrieved documents. More specifically, the decision mechanism is focused on identifying when to favour the entry points or homepages of relevant web sites.

3.1 Decision Mechanism

The decision mechanism we use employs two characteristics of the set of retrieved documents, in order to select an appropriate retrieval approach for each query.

The first characteristic is the *usefulness of the hyperlink structure*, which estimates whether there are non-random patterns of hyperlinks within the set of retrieved documents [14]. If we detect such patterns, then we assume that there are clusters of documents about the query topic. Therefore, it is preferred to favour the entry points, or the central nodes of these clusters.

We define the usefulness of the hyperlink structure as the symmetric Jensen-Shannon divergence between two different score distributions. The first one is the content analysis score distribution $S = \{s_i\}$, where s_i is the content analysis score of the document d_i from the set of retrieved documents D. In order to reduce the computational overhead, we consider only the set D^k of the top k ranked documents, according to the distribution $\{s_i\}$. We define the second distribution $U = \{u_i\}$, so as to favour the relevant documents that point to other relevant documents in D:

$$u_i = s_i + \sum_{d_i \to d_j} s_j, \quad d_i \in D^k, d_j \in D$$

where $d_i \to d_j$ denotes that there is a hyperlink from document d_i to document d_j. We normalise both distributions S and U, so that $\sum_{d_i \in D^k} s_i = \sum_{d_i \in D^k} u_i = 1$ and obtain the distributions $S_n = \{sn_i\}$ and $U_n = \{un_i\}$ respectively. The usefulness of the hyperlink structure is defined as the symmetric Jensen-Shannon divergence $L(S_n, U_n)$ between S_n and U_n, as follows:

$$L(S_n, U_n) = \sum_{d_i \in D} un_i \log_2 \frac{un_i}{\frac{un_i}{2} + \frac{sn_i}{2}} + \sum_{d_i \in D} sn_i \log_2 \frac{sn_i}{\frac{un_i}{2} + \frac{sn_i}{2}} \tag{6}$$

The second characteristic of the set of retrieved documents is a novel estimate of the number of *potential homepages* with all the query terms in the anchor text of their incoming hyperlinks. We assume that if the user submits a query, where all the terms appear in the anchor text of hyperlinks pointing to a homepage of a web site, then it is more useful to favour the homepage as the entry point for the site.

The set of potential homepages H corresponds to the documents with root, top directory, or path URL types, as defined by Kraaij et al. [12]. If we denote the anchor text terms of a document d_i by a_i, and the set of query terms by q, then the number ph_{anchor} of potential homepages with all the query terms in anchor text is defined as follows:

$$ph_{anchor} = |\{d_i | d_i \in (D \cap H) \wedge q \subseteq a_i\}| \tag{7}$$

Our decision mechanism employs $L(S_n, U_n)$ and ph_{anchor} as shown in Table 3. More specifically, if both $L(S_n, U_n)$ and ph_{anchor} are lower or equal to the thresholds t_L and t_{ph} respectively (case I), then we assume that the query is specific and we do not favour the entry points or homepages of web sites. On the other hand, if both $L(S_n, U_n)$ and ph_{anchor} are higher than the thresholds t_L and t_{ph} respectively (case IV), then we assume that it is

	$ph_{anchor} \leq t_{ph}$	$ph_{anchor} > t_{ph}$
$L(S_n, U_n) \leq t_L$	case I (do not favour entry points)	case III (low confidence)
$L(S_n, U_n) > t_L$	case II (low confidence)	case IV (favour entry points)

Table 3: The decision mechanism that selects an appropriate retrieval approach for each query.

more useful to favour the entry points or homepages of web sites, from the set of retrieved documents. For the two other cases, we cannot say with confidence whether we should favour the entry points from the set of retrieved documents.

3.2 Description of experiments and results

We have submitted five official runs for the mixed query task. For all submitted runs, we have indexed the .GOV test collection by removing standard stop-words and applying Porter's stemming algorithm. For the content analysis, we have used the weighting model PL2, as described in Section 2 and Table 1. The term frequency normalisation parameter was automatically set equal to c=1.28, using the approach described in Section 2.

We have used two different retrieval approaches. For the first one (CA), we extend the documents by adding the anchor text of their incoming hyperlinks, and perform content analysis with PL2. For the second approach (CAU150), we re-rank the top 150 documents retrieved with CA, using the score:

$$score_i = s_i \times \frac{1}{\log_2(urlpath\_len_i + 1)} \tag{8}$$

where s_i is the score assigned to document d_i by the approach CA, and $urlpath\_len_i$ is the length in characters of the URL path of d_i.

For both retrieval approaches CA and CAU150, the content analysis scores of documents are increased by a given percentage if the query terms appear either in the anchor text, or in the title of the documents. The percentage of the increase was set empirically, using training data from the TREC2003 topic distillation and known item tasks [6]. More specifically, if we apply CA and a query term t appears in the anchor text or in the title of a document, then we increase the term's weight in the document's score by 8% or 7% respectively. If we apply CAU150 and a query term t appears in the anchor text of a document, then we increase the weight of t in the document's score by 20%.

The evaluation results of our official submitted runs for all topics, as well as for each type of topics, are shown in Table 4. The evaluation measures are the mean average precision (MAP), success at 1 retrieved document (Suc@1), success at 5 retrieved documents (Suc@5) and success at 10 retrieved documents (Suc@10). More specifically, for the named page finding and homepage finding topics, average precision corresponds to the reciprocal of the rank of the relevant retrieved document, when there is one relevant document. The bold entries in Table 4 correspond to the run which resulted in the highest value of the respective evaluation measure.

The first two runs, uogWebCA and uogWebCAU150, correspond to our baselines, where we apply CA or CAU150 for all queries, respectively. With respect to MAP from Table 4, CA is more effective for named page finding queries, while CAU150 is more effective for topic distillation queries. Their performance is similar for homepage finding queries, while CA is more effective than CAU150 over all queries.

For the next three runs, we use the decision mechanism, where the thresholds are set after training with the TREC2003 topic distillation and known item topics. More specifically, in the third run, uogWebSelAn, we use only ph_{anchor}, as shown in Table 5, and apply CAU150 when there are more than $t_{ph} = 1$ potential homepages with all the query terms in the anchor text, otherwise we apply CA. We can see from Table 4 that this run results in

Run	MAP	Suc@1	Suc@5	Suc@10	MAP	Suc@1	Suc@5	Suc@10
	All topics				Topic distillation topics			
uogWebCA	0.4325	0.3733	0.6889	0.7689	0.1280	0.1733	0.5200	0.6667
uogWebCAU150	0.3478	0.3378	0.7111	**0.8444**	**0.1791**	**0.5067**	**0.7733**	**0.8933**
uogWebSelAn	**0.4576**	**0.4444**	**0.7600**	0.8178	0.1655	0.3600	0.6800	0.7733
uogWebSelL	0.3895	0.3467	0.7289	0.8089	0.1625	0.3733	0.6933	0.7867
uogWebSelAnL	0.4569	0.4267	0.7422	0.8000	0.1521	0.2933	0.6267	0.7200
	Named page finding topics				Homepage finding topics			
uogWebCA	**0.6082**	**0.4933**	**0.7867**	**0.8400**	0.5613	0.4533	0.7600	0.8000
uogWebCAU150	0.3324	0.1333	0.6133	0.8133	0.5318	0.3733	0.7467	0.8267
uogWebSelAn	0.6042	**0.4933**	**0.7867**	**0.8400**	0.6031	0.4800	**0.8133**	**0.8400**
uogWebSelL	0.4279	0.2400	0.6933	0.8000	0.5780	0.4267	0.8000	**0.8400**
uogWebSelAnL	**0.6082**	**0.4933**	**0.7867**	**0.8400**	**0.6104**	**0.4933**	**0.8133**	**0.8400**

Table 4: Evaluation of the official submitted runs to the mixed query task of the Web track.

$ph_{anchor} \leq 1$	$ph_{anchor} > 1$
apply CA	apply CAU150

Table 5: The decision mechanism used in run uogWebSelAn.

the highest MAP, and success at 1 and 5 retrieved documents, over all queries. Moreover, it performs similarly to the baselines for the topic distillation and named page finding queries, while it outperforms both CA and CAU150 for the homepage finding queries.

The fourth run, uogWebSelL, is based on a decision mechanism that employs the usefulness of the hyperlink structure $L(S_n, U_n)$, computed from the top $k = 150$ retrieved documents (Table 6). If $L(S_n, U_n)$ is higher than the threshold $t_L = 0.26$, then we apply CAU150, otherwise we apply CA. Considering MAP from Table 4, we can see that this approach works well for the topic distillation and the homepage finding topics, but it is not equally effective for the named page finding topics. If we consider all queries, the run uogWebSelL performs similarly to the baseline uogWebCAU150.

For the fifth run, uogWebSelAnL, we select an appropriate retrieval approach based on both ph_{anchor} and $L(S_n, U_n)$, as shown in Table 7. More specifically, we apply CAU150 if $L(S_n, U_n) > 0.26$ and $ph_{anchor} > 1$, otherwise we apply CA. This run performs as well as the best one, uogWebSelAn, with respect to MAP from Table 4. In addition, it is the most effective for the homepage finding topics and equally effective as applying CA uniformly for named page finding topics.

Overall, we can see from the results in Table 4 that the selective application of different retrieval approaches is more effective than the uniform application of one retrieval approach for all queries. The decision mechanism that employs ph_{anchor} is the most effective over all queries. In addition, the decision mechanism that employs both ph_{anchor} and $L(S_n, U_n)$ performs similarly well. Moreover, it is the most effective approach for both named page and homepage finding queries. In both cases, the textual information from the anchor text is an important source of evidence for selecting an appropriate retrieval approach per query.

$L(S_n, U_n) \leq 0.26$	$L(S_n, U_n) > 0.26$
apply CA	apply CAU150

Table 6: The decision mechanism used in run uogWebSelL.

	$ph_{anchor} \leq 1$	$ph_{anchor} > 1$
$L(S_n, U_n) \leq 0.26$	apply CA	apply CA
$L(S_n, U_n) > 0.26$	apply CA	apply CAU150

Table 7: The decision mechanism used in run uogWebSelAnL.

4 Robust Track

In our participation in the Robust Track, we aim to test a series of techniques, including two novel pre-retrieval query performance predictors, a refined weighting function recommender (WFR) mechanism and an enhanced term frequency normalisation parameter tuning method. In the remainder of this section, we introduce these techniques in Sections 4.1, 4.2 and 4.3, respectively. We also provide the experimental setting in Section 4.4 and describe our runs in Section 4.5.

4.1 Pre-retrieval Query Performance Predictors

For the query performance prediction, we applied two newly proposed predictors, namely the average inverse collection term frequency (AvICTF) and the standard deviation of idf (σ_{idf}). Unlike the state-of-the-art predictors, such as clarity score [7] and query difficulty [3], the computation of these pre-retrieval predictors does not involve the use of relevance scores. As a consequence, the cost of computing these predictors is marginal. The two applied predictors are the following:

- **Average inverse collection term frequency (AvICTF)**. Intuitively, the performance of a query can be reflected by the average quality of its composing terms. To represent the quality of a query term, instead of idf, we apply Kwok's inverse collection term frequency (ICTF). In [13], Kwok suggested that ICTF can be a good replacement for idf which indicates the quality of a query term t. In our work, we use the average of the ICTF values of the composing query terms to infer the overall quality/performance of a query:

  $$AvICTF = \frac{\log_2 \prod_{t \in Q} ICTF}{ql} = \frac{\log_2 \prod_{t \in Q} \frac{token_c}{F}}{ql} \tag{9}$$

  In the above formula, F is the number of occurrences of a query term in the whole collection and $token_c$ is the number of tokens in the whole collection. ql is the length of a given query Q.

- **Standard deviation of idf (σ_{idf})**. This predictor is defined as the standard deviation of the idf of the composing query terms, where idf is given by the INQUERY's idf formula [1]:

  $$idf = \frac{\log_2 (N + 0.5)/N_t}{\log_2 (N + 1)} \tag{10}$$

  where N_t is the number of documents in which the query term t appears and N is the number of documents in the whole collection.

  The assumption behind this predictor is that the composing terms of a poorly-performing query tend to have similar idf values. This indicates that idf fails to differentiate the informative query terms from the non-informative ones, resulting in poor performance.

  According to our work in [11], σ_{idf} has significant linear and Spearman's correlations with average precision on the collection used in this track.

4.2 Weighting Function Recommender Mechanism

The weighting function recommender (WFR) mechanism refines our last year's model selection mechanism [9]. The idea of WFR is to cope with the poorly-performing queries by recommending the optimal weighting functions, including document weighting and term weighting (query expansion) functions, from a set of candidate weighting functions on a per-query basis. The mechanism follows the steps listed below:

1. Using a specific clustering algorithm, cluster a set of training queries into k clusters. The clustering process is based on the above two proposed query performance predictors, i.e. AvICTF and σ_{idf}.
2. Associate the optimal document weighting and term weighting functions to each cluster of training queries by relevance assessment (in this track, we use all the 11 document weighting functions and the 4 term weighting functions, listed in Tables 1 and 2, as the candidate weighting functions).
3. For a given new query, allocate the closest cluster to the query, and apply the associated optimal weighting functions of the allocated cluster.

For the query clustering, we adopt the CURE algorithm [8]. In the CURE algorithm, initially, each element is an independent cluster. The similarity between two clusters is measured by the cosine similarity of the two closest elements (having the highest cosine similarity), where the two elements come from each cluster respectively. If we have n elements to be processed, we start with n clusters. Then, we merge the closest pair of clusters (according to the cosine similarity measure) as a single cluster. The merging process is repeated until it results in k clusters. Here the number k of clusters is the halting criterion of the algorithm.

4.3 Term Frequency Normalisation Parameter Tuning

As mentioned in Section 2, the term frequency normalisation parameter tuning method proposed in [10] uses a set of real queries as training queries. In our participation in this year's TREC, these training queries were obtained using a novel query simulation method that follows the steps listed below:

1. Randomly choose a seed-term from the vocabulary.
2. Rank the documents containing the seed-term using a specific document weighting function.
3. Extract the $exp\_term - 1$ most informative terms from the exp_doc top-ranked documents using a specific term weighting/query expansion function. exp_term is the required number of composing terms of the generated query. exp_doc is a parameter of the applied query expansion methodology, as described in Section 2.
4. To avoid selecting a junk term as the seed-term, we consider the most informative one of the extracted terms in step 3 as the new seed-term.
5. Repeat steps 2 and 3 to extract the $exp\_term - 1$ most informative terms from the exp_doc top-ranked documents, which are ranked according to the new seed-term.
6. The sampled query consists of the new seed-term and the $exp\_term - 1$ terms extracted in Step 5.

Adopting the above query simulation method, our tuning method does not involve the use of real queries.

4.4 Experimental Setting

In this track, there are 249 test topics in total. More specifically, there are 200 old topics used in last year's Robust Track and 49 new topics. Also, from the 200 old topics, 50 poorly-performing topics are chosen as the hard topics.

In our submitted runs, we experimented with three types of queries with respect to the use of different topic fields. The three types of queries are:

- **Short queries**: Only the title field is used.
- **Normal queries**: Only the description field is used.
- **Long queries**: All the three fields (title, description and narrative) are used.

All the applied document weighting and term weighting (query expansion) functions were chosen from the DFR models introduced in Section 2.

For the weighting function recommender (WFR) mechanism, all the 11 DFR document weighting functions and the 4 DFR term weighting functions, listed in Tables 1 and 2, are used as the candidate weighting functions.

For the query simulation of our term frequency normalisation parameter tuning method described in Section 4.3, we applied PL2 and Bo1 weighting functions. We simulated 200 queries to sample the document length distribution of the collection. Using the tuning method, the obtained parameter settings are $c = 5.90$ for short queries, $c = 1.61$ for normal queries and $c = 1.73$ for long queries.

In all our experiments, automatic stop-word removal and Porter's stemming algorithm were applied.

Query expansion was applied in all our experiments. Using a given term weighting model, we extract the 40 most informative terms from the 10 top-ranked documents.

4.5 Description of Experiments

We submitted 10 runs in this track. Among the submitted runs (see Table 8 for run ids and more details):

- We submitted three runs for short queries. AvICTF is applied in all these runs for query performance prediction. uogRobSBase is the baseline for short queries runs. The applied document weighting and term weighting functions are PL2 and Bo1, respectively. Compared to this baseline, uogRobSWR5 and uogRobSWR10 aim to test the weighting function recommender (WFR) mechanism. The threshold setting of WFR, i.e. the number of clusters, is set to 5 for uogRobSWR5 and 10 for uogRobSWR10.

- Our experiments for normal queries are similar. uogRobDBase is the baseline, and WFR is applied in uogRobDWR5 and uogRobDWR10 with the use of different threshold settings (i.e. 5 and 10 respectively). However, I(n)L2 and CS are chosen as the baseline weighting models. AvICTF and σ_{idf} are applied in uogRobDWR10 and the other two, respectively.

- For long queries, besides of WFR, our term frequency normalisation parameter tuning method is also tested. According to our study in [10], this method outperforms the default setting for normal and long queries, and provides comparable performance with the default setting. We compare the tuning method to the use of a default setting that is applied in uogRobLBase. Note that the tuning method is applied in all the runs except this baseline. uogRobLBase uses PL2 and Bo1, respectively. The use of the tuning method differs uogRobLT from uogRobLBase. The other two runs, uogRobLWR5 and uogRobLWR10, are again proposed to evaluate WFR.

Run id	docW function	termW function	c	Predictor
Short Queries				
uogRobSBase	PL2	Bo1	$c = 5.90$	AvICTF
uogRobSWR5	WFR	WFR	$c = 5.90$	AvICTF
uogRobSWR10	WFR	WFR	$c = 5.90$	AvICTF
Normal Queries				
uogRobDBase	I(n)L2	CS	$c = 1.61$	$\gamma 1$
uogRobDWR5	WFR	WFR	$c = 1.61$	$\gamma 1$
uogRobDWR10	WFR	WFR	$c = 1.61$	AvICTF
Long Queries				
uogRobLBase	PL2	Bo1	$c = 1$	AvICTF
uogRobLT	PL2	Bo1	$c = 1.73$	$\gamma 1$
uogRobLWR5	WFR	WFR	$c = 1.73$	$\gamma 1$
uogRobLWR10	WFR	WFR	$c = 1.73$	AvICTF

Table 8: The submitted runs to the Robust track. Query expansion is applied for all the runs. docW function and termW function stand for the applied document weighting function and term weighting function respectively. The applied setting of parameter c for run uogRobLBase, i.e. $c = 1$, is the default setting. *WFR* stands for the weighting function recommender mechanism.

Run id	pre@10	MAP	MAP(X)	#norel
Old queries				
uogRobSBase	.4400	.2826	.0087	32
uogRobSWR5	.4455	.2911	.0072	35
uogRobSWR10	.4605	.2961	.0097	32
New queries				
uogRobSBase	.4816	.3482	.0265	7
uogRobSWR5	.4571	.3272	.0176	8
uogRobSWR10	.4531	.3216	.0215	6
Hard queries				
uogRobSBase	.2640	.1237	.0030	14
uogRobSWR5	.2780	.1305	.0013	15
uogRobSWR10	.3160	.1360	.0025	13
All queries				
uogRobSBase	.4482	.2955	.0098	39
uogRobSWR5	.4478	.2982	.0075	43
uogRobSWR10	.4590	.3011	.0106	38

Table 9: Results of the runs for short queries for the official runs in the Robust track.

Among the document weighting and term weighting functions introduced in Section 2, we have used the optimal ones for the 200 old queries in the baselines.

Tables 9, 10 and 11 summarise the experiment results for short, normal and long queries, respectively. Also, Table 12 provides the obtained Kendall's tau of our predictors with average precision. From the results, we have the following observations:

- In general, WFR achieves higher mean average precision (MAP) than the baselines for the old queries,

Run id	pre@10	MAP	MAP(X)	#norel
Old queries				
uogRobDBase	.4305	.2732	.0062	38
uogRobDWR5	.4460	.2822	.0070	31
uogRobDWR10	.4535	.2861	.0072	32
New queries				
uogRobDBase	.5510	.3888	.0259	6
uogRobDWR5	.5408	.3834	.0234	6
uogRobDWR10	.5286	.3736	.0227	6
Hard queries				
uogRobDBase	.3000	.1230	.0033	15
uogRobDWR5	.3040	.1328	.0032	10
uogRobDWR10	.2960	.1308	.0019	14
All queries				
uogRobDBase	.4542	.2959	.0070	44
uogRobDWR5	.4647	.3021	.0079	37
uogRobDWR10	.4683	.3033	.0083	38

Table 10: Results of the runs for normal queries for the official runs in the Robust track.

Run id	pre@10	MAP	MAP(X)	#norel
Old queries				
uogRobLBase	.4715	.2927	.0130	31
uogRobLT	.4705	.2970	.0136	31
uogRobLWR5	.4800	.3028	.0134	26
uogRobLWR10	.4815	.3084	.0133	25
New queries				
uogRobLBase	.4939	.3586	.0325	3
uogRobLT	.5000	.3776	.0390	2
uogRobLWR5	.5122	.3703	.0388	2
uogRobLWR10	.5143	.3679	.0295	3
Hard queries				
uogRobLBase	.3100	.1609	.0150	34
uogRobLT	.3240	.1552	.0161	33
uogRobLWR5	.3180	.1608	.0158	28
uogRobLWR10	.3120	.1571	.0148	28
All queries				
uogRobLBase	.4759	.3056	.0150	34
uogRobLT	.4763	.3128	.0161	33
uogRobLWR5	.4863	.3161	.0158	28
uogRobLWR10	.4880	.3201	.0148	28

Table 11: Results of the runs for long queries for the official runs in the Robust track.

including the hard queries, but not for the new queries. This might be due to the use of large threshold values for the query clustering process. We are in the process of running unofficial runs with the use of smaller threshold settings. We will report these unofficial runs in the final proceedings.

Run id	Predictor	tau
Short queries		
uogRobSBase	AvICTF	0.259
uogRobSWR5	AvICTF	0.257
uogRobSWR10	AvICTF	0.270
Normal queries		
uogRobDBase	σ_{idf}	0.258
uogRobDWR5	σ_{idf}	0.259
uogRobDWR10	AvICTF	0.240
Long queries		
uogRobLBase	AvICTF	0.163
uogRobLT	σ_{idf}	0.166
uogRobWR5	σ_{idf}	0.172
uogRobWR10	AvICTF	0.176

Table 12: The Kendall's tau of the applied predictors with average precision for the official runs in the Robust track.

- For the new queries, it is interesting to see that using normal and long queries, WFR leads to higher pre@10, but lower MAP than the baselines.
- Our term frequency normalisation parameter tuning method outperforms the baseline in the experiments for long queries. Compared with the baseline, i.e. uogRobLBase, uogRobLT achieves 5.30% of improvement for the new queries, and 2.36% of improvement for all the 249 queries (see Table 11).
- According to the results in Table 12, the obtained Kendall's tau values of our query performance predictors with average precision are not as good as expected, although the correlations for short and normal queries are still respectable. We suggest that this might be due to the use of pseudo query expansion in our runs, which could affect the effectiveness of the applied predictors. We will investigate this issue and report related results in the final proceedings.

5 Terabyte Track

In the Terabyte track, we use Terrier in a distributed setting, inspired by our simulation study in [4]. We test the effectiveness of techniques such as the use of anchor text, pseudo query expansion, and the automatic parameter tuning of term frequency normalisation, for an adhoc retrieval task and the .GOV2 test collection. Moreover, we use a selection mechanism, which allocates the optimal document ranking and query expansion models on a per-query basis. In the remainder of this section, we describe the indexing process and our retrieval experiments.

5.1 Indexing

In order to index the .GOV2 test collection, we employ a local inverted file approach [15]. We split the collection in a number of disjoint sets of documents and index them separately. While indexing, we remove standard stopwords and apply the first step of Porter's stemming algorithm. For each disjoint set of documents, we create the following data structures:

	without anchor text	with anchor text
Total size	17.48GB	18.29GB
Inverted file size	7.77GB	8.47GB
Direct file size	7.00GB	7.70GB
Lexicon size	1.84GB	1.25GB
Document index size	0.87GB	0.87GB

Table 13: The total sizes of the all the data structures, the inverted files and the direct files on disk, with or without anchor text.

- a *direct file* that contains all the terms of each document. The direct file is used for the pseudo query expansion models, given in Table 2.
- an *inverted file* that contains all the document identifiers, in which a term appears.
- a *lexicon* that contains the vocabulary of the indexed documents.
- a *document index* that contains information about the indexed documents.

The direct and inverted files are compressed using γ encoding for the differences of term and document identifiers respectively, and unary encoding for the within-document and within-collection frequencies. The sizes of the data structures on disk are shown in Table 13. Although we index the full text of all documents, the use of compression results in great savings of disk space. More specifically, when we index the content of documents only, the total size of the data structures on disk is 17.48GB, which corresponds to less than 5% of the collection size. In the same index, the total size of the inverted files is 7.77GB, or 1.82% of the collection size. In order to apply pseudo query expansion efficiently, we also built a global lexicon for the whole collection, the size of which is 0.60GB.

Using the same indexing approach, we index the collection a second time, after adding to the documents the anchor text of the incoming hyperlinks. We have added the anchor text from 361,379,741 hyperlinks, without using the information about duplicate documents, or redirects between documents. From Table 13, we can see that the total size of the data structures on disk is 18.29GB, or 4.29% of the collection size, while the total size of the inverted files only is 8.47GB (1.99% of the collection size).

For indexing the collection, we used one AMD Athlon 1600 processor, running at 1.4GHz and one Intel Xeon processor, running at 2.8GHz. The total cumulative CPU time required for building each of the indices was 12,037 minutes and 30,104 minutes respectively.

5.2 Description of Experiments

For our experiments in the adhoc retrieval task of the Terabyte track, we have used a distributed version of Terrier. In this system, a central broker receives the queries and submits them to several independent query servers. The query servers assign scores to documents and send the partial lists of results back to the broker. The broker collects all the partial lists of results and merges them in order to create a final ranked list of retrieved documents. The scores of documents are computed using global statistics, collected by the broker from the query servers. Therefore, the results of our distributed retrieval system are equivalent to the results we would obtain if we used Terrier in a centralised setting.

We have tested both short and long queries. The short queries were created from the title field of the topics, while the long queries were created from all fields of the topics (title, description and narrative).

In Table 14, we present an overview of our official submitted runs. For all five runs, the only parameter of the system, related to the term frequency normalisation, was automatically set to $c = 15.34$ for short queries and

Run	Description	Query Type	Time to retrieve 20 docs.
uogTBBaseS	PL2 content retrieval	short	4 sec
uogTBBaseL	PL2 content retrieval	long	28 sec
uogTBQEL	Pseudo query expansion	long	46 sec
uogTBAnchS	PL2 content and anchor text retrieval	short	3 sec
uogTBPoolQEL	Weighting model selection	long	46 sec

Table 14: Description of our official submitted runs to the Terabyte track.

$c = 2.16$ for long queries, using the approach described in Section 2, with the sampling of queries described in Section 4.3.

Our first run, uogTBBaseS is a content-only baseline, where we employ short queries and assign scores to documents using the weighting model PL2 from the DFR framework, as described in Section 2 and Table 1. For the second run, uogTBBaseL, we use the weighting model PL2 with long queries. In the third run, we employ pseudo query expansion. More specifically, we expand the original query by adding the 20 most informative terms from the 5 top-ranked documents, using the term weighting model Bo1 from Table 2. In the fourth run, uogTBAnchS, we extend documents by adding the anchor text of their incoming hyperlinks, and use short queries for retrieval with PL2. For the last run, uogTBPoolQEL, we used a simple pooling technique to select the appropriate weighting models on a per-query basis. We consider 8 document weighting models from Table 1 (i.e. all the weighting models apart from BB2, PB2 and I(F)B2), and the 4 term weighting models from Table 2, in order to create the pool. Thus, we have $8 \times 4 = 32$ pairs of document weighting and term weighting models. For a given query, we create a pool, which contains documents retrieved among the top 15 ranks by at least 28 pairs of models. Then, we apply the weighting models that retrieve most of the documents in the pool.

In all related experiments, we used 4 machines, with 8 processors and 6GB of memory in total. The configuration of the machines is the following:

- one machine with 2GB of memory and 4 Intel Xeon processors at 2.8GHz.
- one machine with 2GB of memory and 2 AMD Athlon processors at 1.4GHz.
- two machines with 1GB of memory and one Intel Pentium 4 at 2.4GHz.

All the data structures were saved on a RAID disk, mounted on the first machine. The time to retrieve the top 20 documents for each of the five runs is shown in Table 14. It should be stressed that a better throughput could be achieved by using more query servers, as suggested in [4].

6 Conclusions

We have participated in the Web, the Robust and the Terabyte tracks of TREC2004, using our retrieval system, Terrier, in both a centralised and a distributed setting.

In our experiments for the Web track, we use a decision mechanism that identifies the queries for which to favour the entry points of relevant web sites and applies an appropriate retrieval approach. From our results, we can see that using the decision mechanism results in important improvements over the uniform application of one retrieval approach for all queries.

For the Robust track, we have proposed two novel pre-retrieval performance predictors. We employ these predictors in a weighting function recommender mechanism that selects the optimal weighting function for the poorly-performing queries in an effective way. Furthermore, we have employed a refined approach for automatically setting the value of the term frequency normalisation parameters, without the need of real user queries in the tuning process.

With our participation in the Terabyte track, we have evaluated the scalability of a distributed version of Terrier in handling very large test collections, such as the .GOV2. We have seen that even with very limited resources, we can use Terrier to index and experiment with .GOV2.

Overall, we have seen that Terrier is a scalable and modular framework, which provides parameter-free baselines and it can be used effectively in a variety of different retrieval settings

Acknowledgements

The work for the Web Track and the Terabyte Track is funded by a UK Engineering and Physical Sciences Research Council (EPSRC) project grant, number GR/R90543/01. The project funds the development of the Terrier Information Retrieval framework (url: http://ir.dcs.gla.ac.uk/terrier). The work for the Robust Track is funded by the Leverhulme Trust, grant number F/00179/S. The project funds the development of the Smooth project, which investigates the term frequency normalisation (URL: http://ir.dcs.gla.ac.uk/smooth).

References

[1] J. Allan, L. Ballesteros, J. Callan, and W. Croft. Recent experiments with INQUERY. In *NIST Special Publication 500-236: The Fourth Text REtrieval Conference (TREC-4)*, pages 49–63, Gaithersburg, MD, 1995.

[2] G. Amati. *Probabilistic Models for Information Retrieval based on Divergence from Randomness*. PhD thesis, Department of Computing Science, University of Glasgow, 2003.

[3] G. Amati and F. Crestani. Probabilistic learning by uncertainty sampling with non-binary relevance. In F. Crestani and G. Pasi, editors, *Soft Computing in Information Retrieval: Techniques and Applications*. Physica Verlag, Heidelberg, Germany, 2000.

[4] F. Cacheda, V. Plachouras, and I. Ounis. A case study of distributed information retrieval architectures to index one terabyte of text. *Information Processing & Management*, article in press, 2004.

[5] A. Chowdhury, M. C. McCabe, D. Grossman, and O. Frieder. Document normalization revisited. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 381–382. ACM Press, 2002.

[6] N. Craswell, D. Hawking, R. Wilkinson, and M. Wu. Overview of the TREC 2003 Web Track. In *NIST Special Publication 500-255: The Twelfth Text REtrieval Conference (TREC 2003)*, pages 78–92, 2003.

[7] S. Cronen-Townsend, Y. Zhou, and W. B. Croft. Predicting query performance. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 299–306, Tampere, Finland, 2002.

[8] S. Guha, R. Rastogi, and K. Shim. CURE: an efficient clustering algorithm for large databases. In *Proceedings of the ACM SIGMOD Conference, Seatltle, WA*, pages 73–84, 1998.

[9] B. He and I. Ounis. University of Glasgow at the robust track – a query-based model selection approach for the poorly-performing topics. In *NIST Special Publication 500-255: The Twelfth Text REtrieval Conference (TREC 2003)*, pages 636–645, Gaithersburg, MD, 2003.

[10] B. He and I. Ounis. A study of parameter tuning for term frequency normalization. In *Proceedings of the 12th International Conference on Information and Knowledge Management (CIKM)*, pages 10–16. ACM Press, 2003.

[11] B. He and I. Ounis. Inferring query performance using pre-retrieval predictors. In *Proceedings of the 11th Symposium on String Processing and Information Retrieval (SPIRE 2004)*, pages 43–54. Springer Verlag, 2004.

[12] W. Kraaij, T. Westerveld, and D. Hiemstra. The importance of prior probabilities for entry page search. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 27–34. ACM Press, 2002.

[13] K. L. Kwok. A new method of weighting query terms for ad-hoc retrieval. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 187–195. ACM Press, 1996.

[14] V. Plachouras and I. Ounis. Usefulness of hyperlink structure for query-biased topic distillation. In *Proceedings of the 27th Annual International SIGIR Conference on Research and Developement in Information Retrieval*, pages 448–455. ACM Press, 2004.

[15] Ribeiro-Neto B.A. and R. A. Barbosa. Query performance for tightly coupled distributed digital libraries. In *Proceedings of the third ACM conference on Digital libraries*, pages 182–190. ACM Press, 1998.

[16] A. Singhal, C. Buckley, and M. Mitra. Pivoted document length normalization. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 21–29. ACM Press, 1996.