Terrier Information Retrieval Platform

Iadh Ounis, Gianni Amati*, Vassilis Plachouras, Ben He, Craig Macdonald, and Douglas Johnson

University of Glasgow, Glasgow G12 8QQ, UK,
{ounis, gianni, vassilis, ben, craigm, johnsoda}@dcs.gla.ac.uk

Abstract. Terrier is a modular platform for the rapid development of large-scale Information Retrieval (IR) applications. It can index various document collections, including TREC and Web collections. Terrier also offers a range of document weighting and query expansion models, based on the Divergence From Randomness framework. It has been successfully used for ad-hoc retrieval, cross-language retrieval, Web IR and intranet search, in a centralised or distributed setting.

1 Introduction

Experience has shown that the evaluation and cross-comparison of IR models and methods is best done on a common development platform. Hence, our aim for building the Terrier (Terabyte Retriever) IR platform was to provide a publicly available test-bed for the rapid development of IR applications.

Terrier offers a variety of IR models, based on the Divergence From Randomness (DFR) framework[1]. The DFR framework, which can be seen as a generalisation of Harter's 2-Poisson indexing model, is based on a simple idea: the more the divergence of the within-document term-frequency of a term t from its distribution within the collection, the more the amount of information carried by t in the document. In addition to more than 50 parameter-free DFR models, Terrier offers other IR models, such as tf-idf, BM25 and language modelling.

2 Overview of Terrier

The Terrier platform has been designed to efficiently scale up with the size of document collections, operating in either a centralised or a distributed setting. Its main data structures are the direct index, the document index, the inverted index and the lexicon. The direct index stores the identifiers of terms that appear in each document and the corresponding frequencies. It is used for automatic query expansion, but can also be used for user profiling activities. The document index stores information about the document length and identifier, and a pointer to the corresponding entry in the direct index. The inverted index stores

* Gianni Amati is also affiliated to Fondazione Ugo Bordoni, Italy (gba@fub.it).

[1] More details can be found at http://ir.dcs.gla.ac.uk/terrier/description.html.

the posting lists, while the lexicon stores the collection vocabulary and the corresponding document and term frequencies. An additional data structure stores the collection statistics that are used for document ranking. While indexing, we compress the direct and inverted indices, using gamma and unary encodings.

In Figure 1, we outline the indexing process for a document collection. The double-framed boxes correspond to application-dependent modules. Each document in the collection is tokenised and parsed. Depending on the application, we remove stopwords and apply stemming. In this way, we build the direct and document indices. We also build in-memory temporary lexicons for parts of the collection, in order to reduce the required memory during indexing. These lexicons will be merged later, in order to form the lexicon of the whole collection. Next, the inverted index is built from the existing direct index, document index and lexicon. Finally, we collect statistics about the document collection and update the lexicon with information from the inverted file.

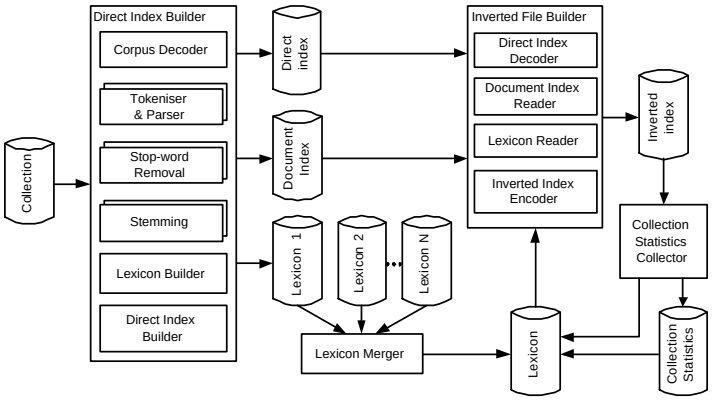

Fig. 1. Indexing process with Terrier.

The retrieval process is outlined in Figure 2. A query is processed by removing stopwords and applying stemming, according to the application requirements. For a given query, Terrier is able to automatically select the optimal document weighting model and/or the appropriate retrieval approaches (e.g. query expansion, anchor text, or link analysis), using among other features, state-of-the-art query performance pre-retrieval predictors. If query expansion (QE) is applied, an appropriate term weighting model is selected and the most informative terms from the top ranked documents are added to the query. Furthermore, Terrier allows to easily fit the retrieval output to the application requirements (e.g. TREC or XML formats), and provides standard evaluation techniques.

Terrier provides a variety of features for indexing and retrieval. First, it uses state-of-the-art compression techniques for data structures. In a distributed setting, a full-text index of the TREC Terabyte track .GOV2 collection (the size of .GOV2 is 426GB) corresponds to only 4.1% of the total collection size (left

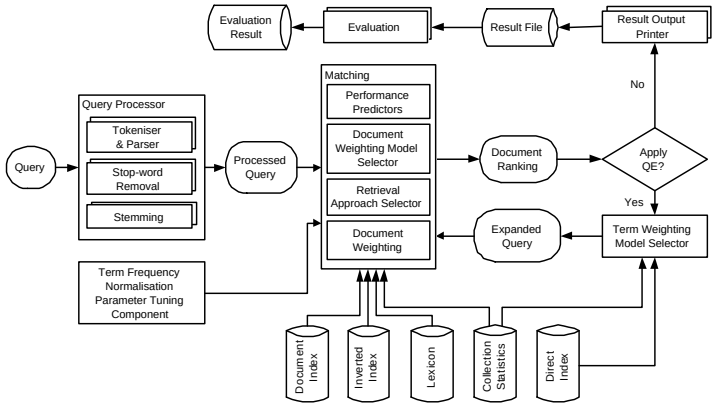

Fig. 2. Retrieval process with Terrier.

part of Table 1). It can also use additional features, such as a retrieval approach selector, position information for proximity and phrasal search, linkage information and HTML features. Terrier provides modular APIs for both indexing and querying, as well as an advanced query language.

Table 1. The size of data structures for a full-text index of the TREC .GOV2 collection, and the evaluation results for the corresponding TREC 2004 Terabyte retrieval task.

Data Structures	Size
All structures	17.48GB
Inverted files	7.77GB
Direct files	7.00GB
Lexicons	1.84GB
Document indices	0.87GB

Run Description	MAP	bpref	P10
Short queries	0.2709	0.3026	0.5306
Short queries + anchor text	0.2690	0.3025	0.5245
Long queries	0.3054	0.3356	0.6327
Long queries + QE	0.3075	0.3359	0.6163
Participants' Median Run	0.1427	0.2015	0.4102

Terrier has been successfully used in the Web, Robust and Terabyte tracks of TREC 2002–04. For the TREC 2004 Terabyte track, Terrier performed significantly better than the median of the participants' submitted runs (right part of Table 1). It has been also used for French and Italian retrieval in CLEF 2003–04, and it is currently used as an intranet search engine for various university and public organisations. A version of Terrier is available for download as open source software from http://ir.dcs.gla.ac.uk/terrier/.

Acknowledgements

This work is funded by a UK Engineering and Physical Sciences Research Council (EPSRC) project grant, number GR/R90543/01.