# WaterlooClarke: TREC 2015 Clinical Decision Support Track

Amira Ghenai[1], Eldar Khalilov[1], Pavel Valov[1], and Charles L. A. Clarke[1]

[1]Department of Computer Science, University of Waterloo, ON, Canada
*aghenai*@uwaterloo.ca, *ekhalilov, pvalov*@gsd.uwaterloo.ca , *claclark*@plg.uwaterloo.ca

October 23, 2015

**Abstract**

Clinical decision support systems help physicians with finding additional information about a particular medical case. In this paper, we develop a clinical decision support system that, based on a short medical case description, can recommend research articles to answer some common medical questions (diagnosis, test and treatment articles). The two different full-text search engines we adopted in order to search over the collection of articles are Terrier and Apache Solr. We test each search engine with different settings and retrieval algorithms. Additionally, we combine the results of the two different search engines using reciprocal rank fusion. The evaluation of the submitted runs using partially marked results of Text Retrieval Conference (TREC) from the previous year shows that the methodologies are promising.

## 1 Introduction

The goal of the current clinical decision support track is to research how documented medical cases can be used when treating patients. When treating a patient, a doctor often needs to investigate additional information in order to determine patient's diagnosis, to select the best treatment plan for a patient with assigned diagnosis, or to decide whether additional testing of a patient is needed.

Sometimes physicians can find necessary information in existing medical research literature. However, considering the amount of published articles available, this investigation might take a considerable amount of time. The goal of clinical decision support systems is to help doctors by associating particular medical research literature with corresponding medical cases, thus making it more accessible. Clinical decision support track attempts to mimic the requirements for development of such systems and to promote their creation along with necessary tools and resources. The main focus of this track is to answer common clinical questions about medical cases, by extracting relevant medical research articles.

## 2 Medical Retrieval System

The two different retrieval search engines here are Terrier and Solr, since we plan to combine the runs using multiple retrieval functions and systems:

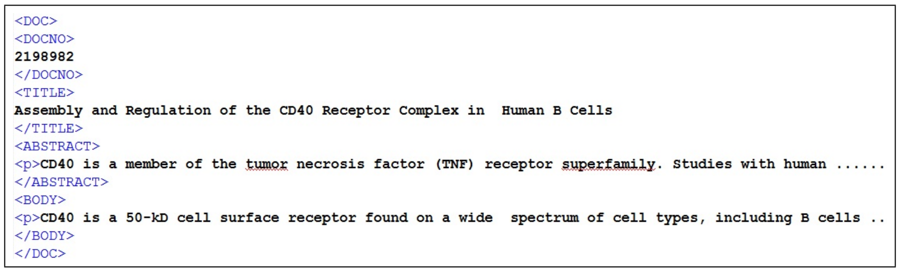
### 2.1 Terrier

We used the Terrier search engine [1] (version 4.0) for the first and third run shown in table 1. Terrier is a highly flexible, efficient, and effective Java open source search engine, readily deplorable on large-scale collections of documents Ounis et al. [2006].

[1]Can be downloaded at http://terrier.org/

#### 2.1.1 Document Pre-processing

Our first step was to prepare a document collection consisting of well-formed XML files: We were interested in only certain fields from the articles tags. Instead of relying on Terrier to extract the needed tags, we decided to transform the articles ourselves and the tags that were included in the documents were: title, abstract, body. We chose those tags because of two main reasons: first, the main information relies in these tags and second, those tags can be identified easily. Both the title and abstract were taken from the documents <front>tag. Under this tag, the title was taken from the <article-title>tag. The abstract was taken from the <abstract>tag. Figure 1 shows an example of an article pre-processed to be indexed by Terrier.



```
<DOC>
<DOCNO>
2198982
</DOCNO>
<TITLE>
Assembly and Regulation of the CD40 Receptor Complex in  Human B Cells
</TITLE>
<ABSTRACT>
<p>CD40 is a member of the tumor necrosis factor (TNF) receptor superfamily. Studies with human ......
</ABSTRACT>
<BODY>
<p>CD40 is a 50-kD cell surface receptor found on a wide  spectrum of cell types, including B cells ..
</BODY>
</DOC>
```

Figure 1: A sample document with PMCID 2198982 after pre-processing and before indexing by Terrier

#### 2.1.2 Text Indexing and Retrieval

We applied Porter stemming algorithm provided by Terrier and we removed stop-words from documents before indexing. Additionally, We indexed and searched only the text documents: No images or videos were considered in our experiments.

After indexing with Terrier, different retrieval functions were performed and the best ones were used in the runs shown in table 1. Terrier provides built-in retrieval functions that we have tested on our articles after tuning the default parameters. The retrieval functions that we chose for our experiments are: BM25, BM25F, PL2 and LnL2 (PL2 and LnL2 achieved similar results that is why we only report PL2). Additionally, We expanded the query by running pseudo-relevance feedback Büttcher et al. [2010] using the top 20 documents retrieved using the initial query. We added a maximum of 10 new query terms to the initial query. Furthermore, we tested the effect of different term score techniques for relevance feedback that Terrier provides. More specifically, we evaluated the effectiveness of the query expansion with Bo1, Bo2 (Bose-Einstein), KL (Kullback-Leibler divergence), CS (chi-square divergence) and Rocchio's algorithm Amati [2003]. Finally, because of the limit of the number of runs (3), we had to pick the best run and submit it. That run is the first one listed in table 1:

- "UWCPL2" Run: PL2 retrieval function was used with Terrier search engine and Rocchio's algorithm for relevance feedback as proven experimentally to be the best method for query expansion.

### 2.2 Solr

We used Apache Solr search engine [2] (version 5.2.1) for the second and third run in table 1. Solr is an open-source full-text search engine, written in Java. For the purpose of full-text indexing and search, Solr utilizes Apache Lucene, an open-source Java search library. Solr provides various search and integration features such as: faceted navigation, hit highlighting, query language, and extensible plugin architecture.

[2]Can be downloaded at http://lucene.apache.org/solr/

### 2.2.1 Documents Pre-processing

We performed a thorough pre-processing of document collection before going through the indexing phase. First of all, we used *standard tokenizer* to split documents into a set of tokens, while treating white space and punctuation as delimiters. Second, we applied *lower case filter* to tokens generated in the previous step. This filter simply converts all upper-case letters to lower-case letters, while ignoring all non-letter symbols. Finally, we applied *porter stemming* algorithm in order to remove common morphological and inflectional endings from tokens. We also tried using less aggressive stemmer called KStem, however this algorithm provided worse results than Porter stemmer.

### 2.2.2 Retrieval Functions

We used two different retrieval functions for Solr: TF-IDF and BM25. As expected, BM25 provided better results than TF-IDF, so we decided to submit the run with the retrieval function BM25 on Solr search engine as it was the best run we could get in terms of precision.

### 2.2.3 Query Pre-processing

When working with Solr, the main effort was spent trying to improve the input query. We tried five different approaches for query modification:

- Filtering the query: We performed a query filtering by keeping only medical terms. This was achieved using publicly available database of medical terms called UMLS Metathesaurus. We were expecting that filtering the query by removing all non-medical terms would increase information per data ratio and improve search accuracy.
  **Example:** A 65-year-old male presents with dyspnea, tachypnea, chest pain on inspiration, and swelling and pain in the right calf.
  **Processed example (filtering):** dyspnea tachypnea chest pain swelling pain in calf.

- Filtering and query expansion: We again filtered the query to leave only medical terms. Later, we expanded the query by adding all possible query terms synonyms from the database.
  **Example:** A young woman in her second gestation presenting with anemia resistant to improvement by iron supplementation, elevated LDH, anisocytosis, poikilocytosis, hemosiderinuria and normal clotting screen.
  **Processed example (filtering + all synonyms):** second gestation anaemia supplementation elevated ldh anisocytosis poikilocytosis secondary gestational anemic elevate anisocytotic.

- Filtering and one term query expansion: We performed a query filtering by keeping only medical terms as the first case. Then, we expanded it using only one synonym from the database that is the most different. We assessed this difference using Levenshtein distance metric.
  **Example:** A 46-year-old woman with sweaty hands, exophthalmia, and weight loss despite increased eating.
  **Processed example (filtering + one synonyms):** sweaty hands exophthalmia increased weight sweatiness gaining ponderal

- Filtering and special query expansion: We filtered the query by leaving only medical terms. Later, we expanded the query by adding all possible query terms synonyms from the database. By adding terms to the query, we forced at least one synonym from every synonym group to be present in returned documents.
  **Example** Young adult woman with 2 weeks of fever and migrating joint inflammation.
  **Processed Example (filtering + all synonyms+boolean operators):** (fever OR pyrectic OR febrility OR pyretic OR febrile OR fevers) AND (joint OR articular OR arthral OR jointedness OR jointed OR joints)

- Filtering and one term special query expansion: We performed a query filtering by keeping only medical terms and expanded it using only one synonym from the database that is the most different. Later, we forced at least one synonym from every synonym group to be present in returned documents.
  **Example:** A 38 year old woman with severe dysmenorrhea, menorrhagia, and menometrorrhagia. PMH of infertility treatment and ectopic pregnancy
  **Processed example (filtering + one synonyms+boolean operators):** (dysmenorrhea OR menorrhalgia) AND (menorrhagia) AND (menometrorrhagia) AND (infertility OR infertile) AND (pregnancy OR cyophoric)

Unfortunately, we could not gain any improvement in terms of accuracy using any of those query modification methods. The best results were achieved only when the input query was left completely unmodified. Table 1 shows the best run we could achieve using Solr search engine which is represented in the second run:

- "$UWCSolrBM25$": The query is pre-processed using the same process as medical articles. We use Standard Tokenizer, Lower Case Filter, and Porter or KStem Stemming Algorithm. Then, BM25 is used as the retrieval function.

### 2.2.4 Summary

To sum up, we tried various combinations of retrieval functions, stemmers and query pre-processing techniques, to find out the best one. We tried two retrieval functions (TF-IDF and BM25), two different stemming algorithms (Porter and KStem), and five different query modification techniques. We found out that the best results are achieved when using BM25 as a retrieval function, Porter Stemming Algorithm for pre-processing of articles and input queries, and without using any query modification techniques.

## 2.3 Rank Fusion

One hypothesis we wanted to test is to combine different retrieval functions to produce better results than having the individual functions run separately. Additionally, we wanted to tested the fusion of different search engine results such as combining Solr and Terrier result sets together. Inspired by Cormack et al. [2009], we decided to use rank fusion that aims to combine multiple ranked document lists (ranks) into one single combined ranked list using reciprocal rank fusion (RRF) algorithm as it generally achieves good, all-round performance. The simple formula used to combine different retrieval functions as cited in Cormack et al. [2009] is as follows:

$$RRFscore(t) = \sum_i \frac{1}{60 + r_i(t)} \quad (1)$$

Where $r_i(t)$ is the rank of the document t in the result set i.

We implemented the RRF algorithm ourselves and tried different combinations of retrieval functions using Terrier and Solr. We tested many combinations such as combining retrieval functions within the same search engine (BM25+PL2, BM25+InL2, PL2+BM25F...) and combining different search engine results together (Terrier_BM25+Solr_BM25, Terrier_PL2+Sol2_BM25 , ...). However, due to a limited number (3) of submissions, only the best run/combination was submitted which is:

- "UWCSolrTerr" Run: PL2 retrieval function was used for Terrier search engine and Rocchio's Algorithm was used as a query expansion technique. In Solr search engine, the original query was expanded, where the query is first filtered to keep medical terms and then expanded with one synonym for each medical term. BM25 was used as the retrieval function for Solr search engine. Reciprocal rank fusion score was finally used to fuse the results of Terrier and Solr by producing a new combined ranked list.

Table 1: Runs Summary

| Run id | Search Engine | Retrieval Function | Description |
|---|---|---|---|
| *UWCPL2* | Terrier | PL2 | Using Rocchio's algorithm for query expansion and PL2 using Terrier search engine |
| *UWCSolrBM25* | Solr | BM25 | The run was generated using Solr search engine and BM25 |
| *UWCSolrTerr* | Solr, Terrier | BM25, PL2 | Rank fusion of Solr results with query expansion and BM25 with Terrier with PL2 |

## 3 Results

The three runs discussed in section 2.1 and 2.2 were submitted. The evaluation metrices used for the clinical decision support track are infAP, infNDCG, R-prec and P@10 which are extended inferred metrics that are discussed in [Yilmaz et al., 2008].

As table 2 shows, using Terrier search engine and PL2 as a retrieval function with Rocchio's algorithm for query expansion outperformed all the other search engine combinations and query expansion techniques. UWCPL2 got the highest P@10 score given across all runs with a value of 0.4300, the highest inferred normalized discounted cumulative gain and R-precision scores (0.2542 and 0.2195 respectively).

Table 2: infAP, infNDCG, R-prec and P@10 scores for all of our runs

| Run | infAP | infNDCG | R-prec | P @ 10 |
|---|---|---|---|---|
| **UWCPL2** | 0.0630 | 0.2542 | 0.2195 | 0.4300 |
| UWCSolrBM25 | 0.0449 | 0.2121 | 0.1793 | 0.3933 |
| UWCSolrTerr | 0.0053 | 0.0655 | 0.0235 | 0.1200 |

These results suggest that the UWCPL2 run was successful for most of the topics using Rocchio's algorithm as query expansion and PL2 with Terrier. We observe that the queries corresponding to these topics were generally long in length (due to strong performance of query expansion). The refinement of these queries by detecting enough medically specific keywords in the query to retrieve highly relevant documents.

There were few topics for which the median of all three of our runs performed consistently poor such as 18, 20, 25, 27 and 28 where P@10 $\approx 0$ and R-prec $\approx 0$. This result reveals few limitations of our approach. Some of these topics were very short and contained very few technical, specific medical nouns. Thus, query expansion technique to expand the base query was not very helpful.

## 4 Conclusion

In this paper, we demonstrated our contribution to the 2015 TREC Clinical Decision Support Track in order to build a system able to recommend research articles and to answer common medical questions. More specifically, We opted for an exploratory approach in which we used different full-text search engines (Terrier and Solr Apache) and we explored the hypothesis of combining different retrieval functions and search engines. In Terrier, we studied the effect of different term score techniques when using pseudo-relevance feedback to expand the input query. For Solr, different retrieval functions were used and various input query expansion techniques were explored. We evaluated our system using TREC results from the previous year and the best runs were selected to be submitted.

## 5 Future Work

Although we believe that we achieved promising results, There is much to be done in future. First of all, pre-processing of medical articles could be done in a different way. Different tokenizers and stemmers can be used for both medical articles and input queries. Second, more retrieval functions can be explored for both Terrier and Solr search engine such as BM25L Lv and Zhai [2011] which is a modified version of BM25 that is known to handle long documents better than other retrieval functions. Third, despite the fact that query modification didn't improve retrieval results in Solr, it still might be possible to achieve some accuracy gain via query expansion using a different approach. Fourth, we can explore different methods of fusing the results of Terrier and Solr search engine rather than RRF technique such as Condorcet Fusion Montague and Aslam [2002].

## References

G. Amati. *Probability models for information retrieval based on divergence from randomness*. PhD thesis, University of Glasgow, 2003.

S. Büttcher, C. L. Clarke, and G. V. Cormack. *Information retrieval: Implementing and evaluating search engines*. Mit Press, 2010.

G. V. Cormack, C. L. Clarke, and S. Buettcher. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 758–759. ACM, 2009.

Y. Lv and C. Zhai. When documents are very long, bm25 fails! In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 1103–1104. ACM, 2011.

M. Montague and J. A. Aslam. Condorcet fusion for improved retrieval. In *Proceedings of the eleventh international conference on Information and knowledge management*, pages 538–548. ACM, 2002.

I. Ounis, G. Amati, V. Plachouras, B. He, C. Macdonald, and C. Lioma. Terrier: A High Performance and Scalable Information Retrieval Platform. In *Proceedings of ACM SIGIR'06 Workshop on Open Source Information Retrieval (OSIR 2006)*, 2006.

E. Yilmaz, E. Kanoulas, and J. A. Aslam. A simple and efficient sampling method for estimating ap and ndcg. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 603–610. ACM, 2008.