BIOINFORMATICS ORIGINAL PAPER Vol. 22 no. 18 2006, pages 2298–2304
doi:10.1093/bioinformatics/btl388

Data and text mining

Text similarity: an alternative way to search MEDLINE

James Lewis, Stephan Ossowski, Justin Hicks, Mounir Errami and Harold R. Garner*

University of Texas Southwestern Medical Center, Eugene McDermott Center for Human Growth and Development, Division for Translational Research, 5323 Harry Hines Boulevard, Dallas, TX 75390, USA

Received on May 22, 2006; revised on July 5, 2006; accepted on July 7, 2006
Advance Access publication August 22, 2006
Associate Editor: John Quackenbush

ABSTRACT

Motivation: The most widely used literature search techniques, such as those offered by NCBI's PubMed system, require significant effort on the part of the searcher, and inexperienced searchers do not use these systems as effectively as experienced users. Improved literature search engines can save researchers time and effort by making it easier to locate the most important and relevant literature.

Results: We have created and optimized a new, hybrid search system for Medline that takes natural text as input and then delivers results with high precision and recall. The combination of a fast, low-sensitivity weighted keyword-based first pass algorithm to cast a wide net to gather an initial set of literature, followed by a unique sentence-alignment based similarity algorithm to rank order those results was developed that is sensitive, fast and easy to use. Several text similarity search algorithms, both standard and novel, were implemented and tested in order to determine which obtained the best results in information retrieval exercises.

Availability: Literature searching algorithms are implemented in a system called eTBLAST, freely accessible over the web at http://invention.swmed.edu. A variety of other derivative systems and visualization tools provides the user with an enhanced experience and additional capabilities.

Contact: Harold.Garner@UTSouthwestern.edu

1 INTRODUCTION

The NCBI's PubMed system is the most widely used method for accessing MEDLINE. PubMed employs a Boolean search strategy in which users enter search terms and logic operators (AND, OR, NOT) to retrieve documents from MEDLINE. Careful indexing allows Boolean systems to return results quickly with minimal computational overhead. The efficiency of Boolean retrieval systems made them the most popular way to retrieve documents from electronic libraries at a time when computer hardware was expensive. However, because of the complexity of index languages and the necessity of using Boolean connectives, literature searches have historically been performed by professional search experts (such as librarians) (Hersh, 2003). It is widely reported that less-experienced searchers, including many that regularly use the PubMed system over the web, do not utilize these systems as effectively as more experienced searchers (Bernstam, 2001; Haynes *et al*., 1990; McKibbon *et al*., 1995). Reasons for this performance gap include failure of searchers to employ the best search terms, and failure to use Boolean logic operators effectively (Wildemuth and Moore, 1995; Bradigan and Mularski, 1989). Other shortcomings of Boolean systems involve differences in term usage between searchers and indexers (Hersh, 2003), and disagreement among indexers as to the correct index terms to apply to a document (Funk and Reid, 1983; Leonard, 1975). Finally, the execution of a successful literature search using a Boolean system has been shown to be a significant burden to the novice searcher, with the average novice searcher (third year medical student) requiring 14 separate queries to attain their objective in one study (Wildemuth and Moore, 1995).

In contrast, in text similarity searching (TSS) a user supplies an 'example document' (such as a paragraph of natural language text) and the search-system returns a set of documents similar to the example (Salton, 1983; Van Rijsbergen, 1979). TSS systems typically represent documents as lists of words and their frequencies of occurrence. These lists of words and counts, referred to as 'word-count vectors' or simply 'vectors', can be compared with one another statistically by a variety of techniques to measure their relative performance. While this similarity computation requires more computer resources than simple Boolean retrieval, TSS searching is generally held to yield better results than Boolean searching (Salton, 1991; Wiesman *et al*., 1997), and has become more practical as the cost of computer resources has fallen precipitously. There is also evidence to reinforce the intuition that there is no performance gap between experienced and inexperienced users of TSS systems (Hersh and Hickam, 1995), because TSS systems require substantially less effort on the part of the user. TSS systems save researchers time and effort by shifting the burden of constructing an effective query from the researcher to the computer system. Finally, TSS systems provide the researchers with more useful information than Boolean systems by displaying the most-relevant literature first, rather than sorting by publication date as is done by many Boolean systems including PubMed.

We demonstrate that our novel text similarity algorithm, when coupled with word-vector approaches, is a highly effective alternative to traditional techniques, enabling us to offer the biomedical research community a literature search tool which is optimized, simple and free.

2 METHODS

2.1 Evaluation techniques

Traditionally the effectiveness of an information retrieval (IR) system is measured in terms of precision and recall. For an IR system returning a set of documents from a library, precision measures the percentage of returned

*To whom correspondence should be addressed.

documents relevant to the query, while recall measures the percentage of all relevant documents in the library returned by the system.

Various TSS search algorithms were implemented and subjected to two evaluation techniques. In the first technique, a group of novice searchers (10 first-year medical students) were asked to evaluate the effectiveness of several algorithms using our TSS system. Each of them composed or selected a paragraph on a topic with which they were familiar, and used that paragraph as a query into MEDLINE. They accessed our system over the internet to search MEDLINE with several different search parameters. They then rated the first 30 results returned by each query as 'relevant' or 'not relevant'.

In the second, more quantifiable approach, we subjected our system to a batch evaluation using resources provided by TREC (Hersh and Bhupatiraju, 2003a; Voorhees, 2003). TREC is an annual workshop designed to foster innovation in information retrieval and information extraction. TREC offers libraries of searchable text, an evaluation set of 50 queries from each year's workshop, and a list of items from the library judged to be relevant to each query. TREC also provides an evaluation tool which measures performance in a variety of ways. Our performance analysis of various algorithms makes use of precision–recall curves and mean average precision (MAP) as reported by this evaluation tool.

For this study, all queries were executed within a research platform called eTBLAST (http://invention.swmed.edu/etblast/index.shtml), a software system in which each of the Boolean, TSS and hybrid combinations are implemented. The current production system available for public use has a simple user interface. The production system currently resides on a 12 node cluster consisting of 3 GHz Intel Pentium Xeon processors running Linux RedHat 9. Within this production environment most user queries return results in <1 min.

2.2 Algorithms

Because alignment algorithms are prohibitively slow for searching a document collection the size of MEDLINE, our implementation works in two phases. In the first stage, a vector-based TSS algorithm is used to identify the top 400 highest scoring matches. These are then re-ranked using text alignment.

Our vector-based TSS algorithms implement approaches previously described in the literature (Salton, 1983; Van Rijsbergen, 1979; Hersh and Bhupatiraju, 2003b). In the preliminary processing step, word-count vectors are created from library documents. Non-essential 'stopwords' are removed from the vector, as they are not useful in the IR process (Salton, 1983). At run-time, a query consisting of a paragraph of natural language text is submitted to the system. The query is immediately converted to a vector representation (stopwords removed), expanded by stemming and then compared with the vector representation of each library document by one of the several similarity functions. The IR literature documents a variety of similarity functions and weighting schemes to be used for the comparison of two documents represented as vectors of words and frequency-counts. A primary aim of this study was to compare these similarity functions and weighting schemes against one another in a controlled environment in order to determine which achieved the best results. This optimized first step can then be connected to our sentence alignment system to form the final complete system.

2.2.1 Similarity functions We implemented three widely cited functions for calculating similarity scores (Salton, 1983; Van Rijsbergen, 1979). Let us define some terms:

n = number of unique words in the library

$$X = (x_1, \ldots, x_n), \text{ where } x_i = \begin{cases} 1 & \text{if word } i \text{ is in query} \\ 0 & \text{if word } i \text{ is not in query} \end{cases}$$

$$Y = (y_1, \ldots, y_n), \text{ where } y_i = \begin{cases} 1 & \text{if word } i \text{ is in library text} \\ 0 & \text{if word } i \text{ is not in library text} \end{cases}$$

X and Y are defined as binary vector representations of the user query and a library document respectively, denoting the presence or absence of a word in either document. Given these definitions, the similarity functions we implemented are

$$\text{Cosine coefficient} = \frac{\sum_{i=1}^{n} x_i \cdot y_i}{\sqrt{\sum_{i=1}^{n} x_i^2 \times \sum_{i=1}^{n} y_i^2}} \quad (1)$$

$$\text{Jaccard coefficient} = \frac{\sum_{i=1}^{n} x_i \cdot y_i}{\sum_{i=1}^{n} x_i + \sum_{i=1}^{n} y_i - \sum_{i=1}^{n} x_i \cdot y_i} \quad (2)$$

$$\text{Dice coefficient} = \frac{2\sum_{i=1}^{n} x_i \cdot y_i}{\sum_{i=1}^{n} x_i + \sum_{i=1}^{n} y_i} \quad (3)$$

The cosine similarity function (CSF) is the most widely reported measure of vector similarity. The virtue of the CSF is its sensitivity to the relative importance of each word (Hersh and Bhupatiraju, 2003b). The Jaccard Coefficient, in contrast, measures similarity as the proportion of (weighted) words two texts have in common versus the words they do not have in common (Van Rijsbergen, 1979). Finally Dice's coefficient simply measures the words that two texts have in common as a proportion of all the words in both texts.

2.2.2 Weighting We implemented and tested three different weighting schemes for measuring the importance of keywords. Each of the similarity functions above can be used in conjunction with each of these weighting schemes to yield nine possible combinations. The first weighting scheme involved a simple word count. In this scoring approach, each word received a score of m if it occurred m times in a text (user query or library document). We refer to this as Term Frequency 1 (TF1) weighting, and we express it by redefining the terms x_i as the number of occurrences of word i in the query and y_i as the number of occurrences of the word i in the library text.

Perhaps the most common word-weighting scheme reported in the IR literature is TF∗IDF weighting. TF∗IDF weighting is described in numerous variants across the literature. Two different TF∗IDF variants were implemented for this study, which we call TF1∗IDF and TF2∗IDF.

Inverse document frequency (IDF) is a measure of the relative rarity of a word in a library, where rarely used words are considered to be more specific. Multiplying the frequency of a word in an individual text by the relative rarity of the word in the library is a way of quantifying the idea that the words which discriminate meaning best are those occurring frequently in a small number of texts. The IDF for each word in MEDLINE was calculated as

$$\text{IDF}_i = \log\left(\frac{\text{number of documents in database}}{\text{number of documents with term } i}\right). \quad (4)$$

The weight of a word then becomes TF1 multiplied by IDF. Thus, our cosine similarity function (1) becomes

$$\text{Cosine coefficient} = \frac{\sum_{i=1}^{n} x_i \cdot y_i \cdot \text{IDF}_i}{\sqrt{\sum_{i=1}^{n} x_i^2 \times \sum_{i=1}^{n} y_i^2}}, \quad (5)$$

where again, x_i is the number of occurrences of word i in the query and y_i is the number of occurrences of the word i in the library text. Jaccard and Dice coefficients are defined similarly. We refer to this as TF1∗IDF weighting.

The concept underlying the second TF∗IDF variant is that, at some threshold, increased repetition of a word in a document does not necessarily imply increased importance. To represent this intuition in a weighting scheme a log function was applied to the term frequency (TF1). The log base 1.6 was chosen to down-weight the scores of words in user queries and MEDLINE abstracts because it does not significantly alter the weights of words occurring from one to four times in a text, while it does down-weight words more significantly when they appear more than five times. Thus our

second term frequency weighting (TF2):

$$x_i = \begin{cases} \log_{1.6} & (\text{number of occurences of word } i \text{ in query}) \\ 0 & \text{if word } i \text{ not in query} \end{cases} \quad (6)$$

$$y_i = \begin{cases} \log_{1.6} & (\text{number of occurences of word } i \text{ in library text}) \\ 0 & \text{if word } i \text{ not in library text} \end{cases} \quad (7)$$

Substituting these definitions into Equation (5) for CSF yields the final set of combinations for weighting schemes and similarity functions, where x_i and y_i are now defined using (6) and (7). These definitions are applied to Jaccard and Dice coefficients similarly. We refer to this as TF2∗IDF weighting.

2.2.3 Text alignment The IR literature describes dynamic programming algorithms for computing the edit distance between two strings (Chen *et al*., 2004). Dynamic programming alignment algorithms are used in the popular BLAST utility (Jurafsky and Martin, 2000) for DNA sequence or protein similarity searching, e.g. however, alignment algorithms are rarely used in large-scale information retrieval because they are relatively slow (Altschul *et al*., 1990). The value of the alignment approach (versus the vector approach) lies in the preservation of word order and sentence order information, which are lost in the vector representation. We implemented a novel alignment algorithm (Yamamoto *et al*., 2003) which performs an alignment between a user's input paragraph and each of a set of documents. This algorithm utilizes a standard substitution matrix in which insertions and deletions are penalized with −1, a mismatch is scored 0 and a match receives its IDF weight. We use a local full text alignment, in which the highest score from any cell in an alignment matrix is taken as the overall alignment-based similarity score.

Two variations of text alignment were investigated. In the first, which we refer to as full text alignment, entire library documents (MEDLINE abstracts) were aligned with user queries. The second algorithm, which we refer to as sentence alignment, aligns individual sentences from a library document with individual sentences from the user query. Summing over the maximum alignment values for each query sentence yields an overall alignment-based similarity score. This alignment-based similarity score, which is an output, was calibrated by measuring the distribution of scores from a set of synthetic abstracts assembled from random sets of keywords. This sentence-level granularity was chosen to compensate for an author's choice of sentence order, while preserving concepts and their relationships within individual sentences. The theoretical run time for the sentence alignment algorithm is the same as the standard alignment algorithm, $O(n^2)$.

2.3 Preprocessing

At the time of this experiment, the MEDLINE database consisted of 49.9 Gb of XML files, which was condensed to 13 Gb of searchable text files by representing each title, abstract and authors list as a word-frequency vector, with stopwords removed. This extensive pre-processing requirement represents a significant commitment of computer resources for a library the size of MEDLINE; however the process is now automated, requiring no additional commitment of human resources. The compute-time required to keep the vector representation of MEDLINE updated on a daily basis is negligible.

3 RESULTS

3.1 Results for user testing of TSS

In our first experiment, 10 novice searchers (first-year medical students) were asked to select or create a paragraph of text on a familiar topic, and use it as a query in our search engine. Students were asked to execute a battery of searches of MEDLINE using our system with the feature settings described in Table 1.

The queries performed by each student returned 30 abstracts ranked by similarity score, and students rated each returned abstract as 'relevant' or 'not relevant'. Our statistical benchmarking of 1000 abstracts (Section 3.3.2) demonstrate that 99.978% of the 30 first results are relevant with a Z-score > 3 explaining our choice for the 30 first results to be considered by each student. The average precision attained by all users for all algorithms was 76.8 ± 18.3, while the highest average precision attained by all users on their best search was 85.0 ± 12.3 (Table 2). Given the restricted size of our user group it was impossible to draw statistically significant conclusions regarding the effectiveness of the various system parameters based solely on user testing.

Table 1. System parameters used by students for queries into MEDLINE

Test	Similarity function	Weighting function
1	Cosine similarity	TF2
2	Cosine similarity	TF2∗IDF
3	Jaccard similarity	TF2∗IDF
4	Full text alignment	

Table 2. Mean and standard deviation of relevant documents retrieved and percent precision for quires by 10 medical students

	Relevant documents retrieved	Percentage precision
TSS Average	23.0 ± 5.5	76.8 ± 18.3
TSS Best	25.5 ± 3.7	85.0 ± 12.3

Ten students each preformed four text similarity searches using our system with various parameters. Students were asked to rate the first 30 results as 'relevant' or 'not relevant'.

3.2 Results for TSS based on word-vectors using TREC evaluation

The TREC routing task from TREC 2 and 3, and the TREC Hard Track from 2003 (Voorhees, 2003), were selected as our test set for a systematic quantitative evaluation of our various TSS algorithms. We will refer to this as our batch query set. These particular sets were selected because their topic descriptions are generally longer than those of other TREC tasks, and thus more appropriate for use in a natural language interface. Taken together, these three topic sets comprise 150 queries, which were executed in batches against word-vectors we created from TREC libraries. Results were computed using the TREC evaluation tool. Table 3 summarizes the TSS strategies we tested.

To test any qualitative domain-specific performance we also developed a unique version of the TREC Genomics Track. (Hersh and Bhupatiraju, 2003b) The Genomics Track clustered MEDLINE records based upon the 50 GeneRIF (gene reference into function) short texts. We cyclically searched each abstract in each cluster against the 525 938 MEDLINE records in the Genomics Track. Since we used each cluster member as a query rather than the inappropriate, short GeneRIFs that served 'as pseudorelevance judgments', the TREC evaluation tool could not be used to compute precision and recall statistics and the data could not be combined directly with the TREC 2 and 3, and the

Table 3. Description of system parameters evaluated using TREC

Test	Similarity function	Weighting function
Similarity Function	Cosine similarity	TF2∗IDF
	Jaccard similarity	TF2∗IDF
	Dice similarity	TF2∗IDF
Weighting Function	Cosine similarity	TF1
	Cosine similarity	TF1∗IDF
	Cosine similarity	TF2∗IDF
Alignment Functions	Full text alignment	
	Sentence alignment	

The cosine similarity function with TF2∗IDF weighting is used as a baseline against which to measure other TSS variants.

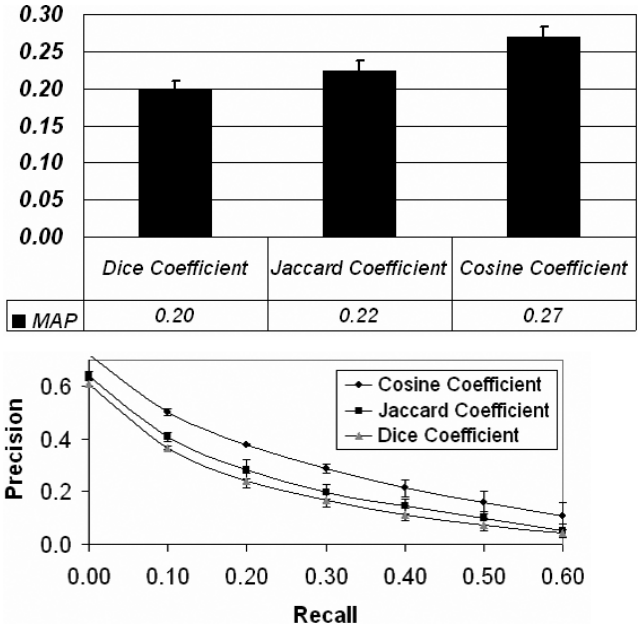

Fig. 1. MAP (top) and precision/recall curves (bottom) for cosine, Jaccard and Dice similarity coefficients weighted with log word-frequency and IDF. Results from 150 queries from TREC routing 2, 3 and Hard Track 2003, calculated with the TREC evaluation tool.

TREC Hard Track data. However, we observed a similar qualitative behavior in the relative performance of each of the similarity and weighting functions tested.

We began by testing the effectiveness of the three similarity functions. We executed the batch query set against the TREC library using the CSF (1), Jaccard (2) and Dice (3) functions with TF2∗IDF weighting. Figure 1 shows that the CSF outperformed both the Dice and Jaccard coefficients on the TREC evaluation, achieving a MAP of 0.27. This means that executing an average query with this function would yield a relevant result for every 3.7 results returned. Using the Jaccard function only one in 4.54 results would be expected to be relevant, while using the Dice coefficient, only one in 5 results would be expected to be relevant. We conclude that the traditional cosine similarity measure achieves better performance than the Dice or Jaccard measures in TSS.

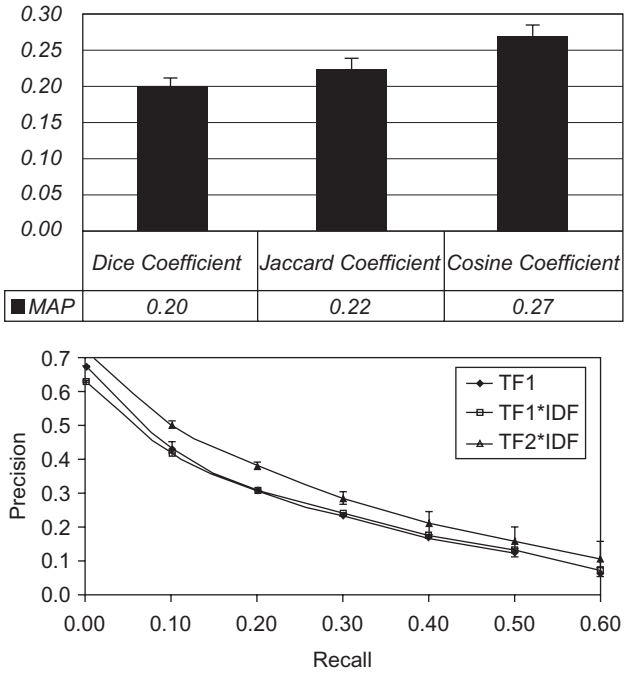

Fig. 2. MAP (top) and precision/recall curves (bottom) for simple word count weighting, IDF weighting and log term-frequency with IDF. The cosine similarity function was used in all tests. Results from 150 queries from TREC routing 2, 3 and Hard Track 2003, calculated with TREC evaluation tool.

3.2.1 Performance of statistical weighting using TREC In order to measure the impact of different weighting schemes on TSS tasks, we ran the batch query set using CSF (1) with three different weighting schemes, TF1, TF1∗IDF (Fig. 2). The TF2∗IDF scheme, with term frequency as a low-base log function of word count, outscored both the TF and TF1∗IDF weighting, with a MAP of 0.27. We conclude that TF2∗IDF weighting, used in concert with cosine similarity, yields the best performance of the traditional TSS methods tested.

3.3 Evaluation of novel TSS algorithms

3.3.1 Performance of alignment matrix evaluated using TREC In order to quantify the performance of alignment algorithms, we executed the batch query set using both full text alignment and our novel sentence alignment algorithm, and compared the results with the baseline described above (CSF with TF2∗IDF weighting). The sentence alignment approach yielded better results than either full text alignment or TSS with word-count vectors and any combination of scoring function or weighting (Fig. 3). Although the theoretical run time of an alignment algorithm is longer than that of TSS using word counts, our implementation significantly reduces (by a factor of ∼37 000, given that MEDLINE contains ∼15 million abstracts) the number of documents which must be aligned by performing an initial search using TSS with word counts and then re-ranking a large number of results by alignment. Thus, the actual execution time of the two approaches is nearly equal. We conclude that our hybrid search system with sentence alignment algorithm is an improvement over traditional methods of TSS,

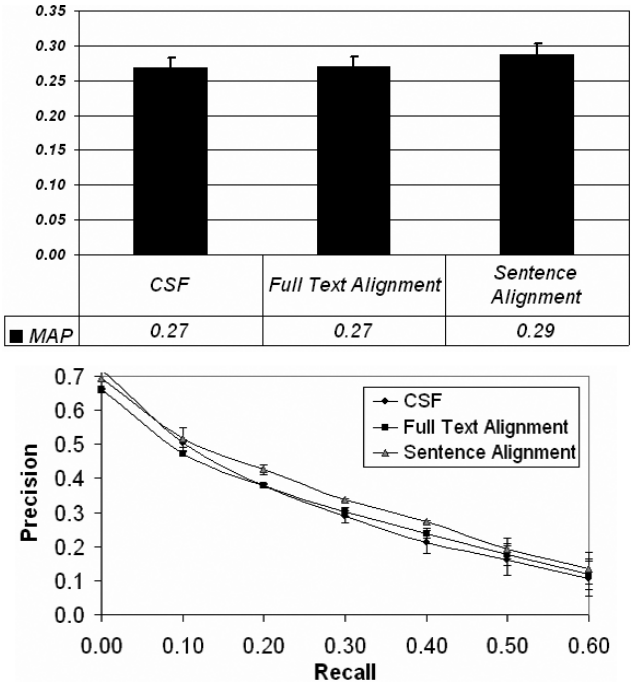

Fig. 3. MAP (top) and precision/recall curves (bottom) for full text alignment and sentence alignment (CSF with TF2∗IDF weighting shown for reference). Results from 150 queries from TREC routing 2, 3 and Hard Track 2003, calculated with TREC evaluation tool.

Table 4. Results of evaluation of TSS functions and weighting schemes on 150 TREC batch queries

Similarity function	Weighting function	MAP
Sentence alignment	n/a	0.29
Full text alignment	n/a	0.27
Cosine similarity	TF2∗IDF	0.27
Cosine similarity	TF1∗IDF	0.24
Cosine similarity	TF1	0.23
Jaccard similarity	TF2∗IDF	0.22
Dice similarity	TF2∗IDF	0.20

Ordered by MAP as reported by TREC evaluation tool.

yielding increased precision without a significant increase in runtime.

Table 4 summarizes the performance of the algorithms tested using standard TREC evaluation. Overall, the sentence alignment algorithm outperformed the standard cosine similarity search with the TF2∗IDF weighting scheme.

3.3.2 Comparison of sentence alignment algorithm with PubMed Related Articles As a supplement to PubMed's Boolean search facility, NCBI offers a Related Articles feature which uses TSS algorithms to compare newly published MEDLINE articles with the MEDLINE library. Related Articles are computed using text similarity search and presented to users as a list of MEDILNE

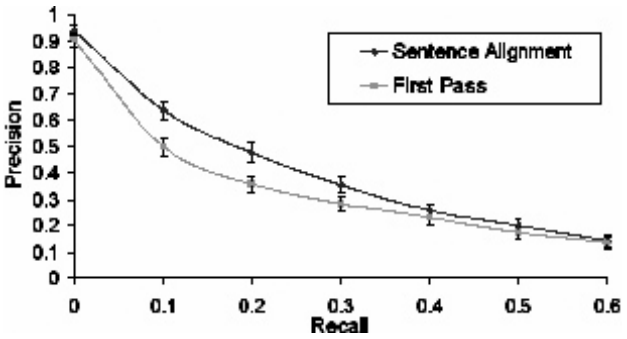

Fig. 4. Precision values are averages ± standard error of the mean ($n = 100$ in each group). The sentence alignment algorithm returns citations comparable with those returned by PubMed's Related Articles. A total of 100 random MEDLINE entries were used as queries into MEDLINE using our sentence alignment algorithm, and the results compared with the top 30 PubMed Related Articles. The lower curve depicts precision and recall of the Cosine Similarity function with TF2∗IDF weighting. The upper curve depicts precision and recall after re-ranking results using our Sentence Alignment approach. At 10% recall, or 3 of the top 30 Related Articles returned, the precision of Sentence Alignment is 64%, versus only 50%. The *p*-value for each data point from 0 to 0.6 is 0.026, 6.95E−08, 8.69E−09, 2.31E−07, 0.005, 0.003, 0.085, respectively.

citations ranked by similarity (Perstemlidis and Garner, 2004). Because our search function also utilizes TSS algorithms (in its first step), we would expect our implementation to provide comparable results given the same inputs.

Only one study has evaluated the relative usefulness of PubMed's Related Articles feature, to our knowledge. A small group of users reported that the usefulness of Related Articles drops off sharply after the first 20–30 ranked results (Bernstam, 2001). Recent research on user interactions with IR systems also indicates that users of internet search engines rarely view more than the top 5 ranked results (http://www.ncbi.nlm.nih.gov/entrez/query/static/computation.html; Granka *et al*., 2004). This research suggests that the relevance of the highest ranked documents is of paramount importance.

We randomly selected 100 MEDLINE entries with abstracts from the past 5 years and used them as queries on our hybrid system that implements the TF2∗IDF Cosine Similarity algorithm followed by our Sentence Alignment algorithm. We then compared the results of our MEDLINE searches with the top 30 Related Articles of each of those queries. Precision/recall curves (Fig. 4) show that the effectiveness of our approach is comparable with that of PubMed's Related Articles. On average, of the top 5 documents retrieved and re-ranked by our Sentence Alignment algorithm, >60% were among the top 30 Related Articles. This shows that our Sentence Alignment algorithm does a good job of returning relevant articles in the critical top 5 positions. It should be noted that within the first 400 documents returned by the first stage TF2∗IDF Cosine Similarity algorithm, on average 44% of all Related Articles are found, placing an upward bound on the expected performance of the hybrid system. The fact that our Sentence Alignment algorithm shows higher precision at all levels of recall supports the claim that our novel Sentence Alignment re-ranking improves upon the Cosine Similarity approach by ranking relevant results more highly than they are ranked by Cosine Similarity alone. We conclude that our implementation performs a service comparable with PubMed's

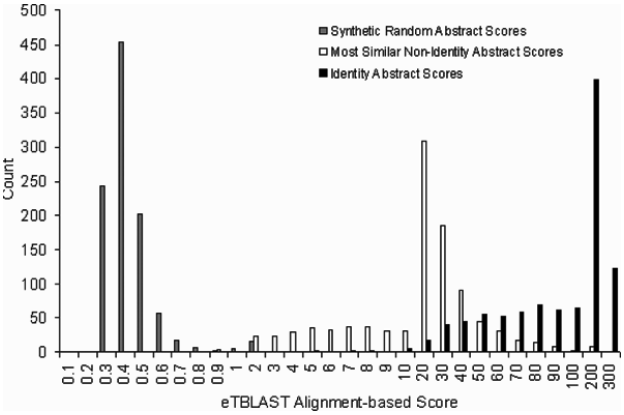

Fig. 5. Analysis of synthetic queries designed to resemble the size and word use distribution of MEDLINE abstracts and a set of MEDLINE abstracts selected at random were used to compute the distribution of top alignment-based scores returned from a search against MEDLINE.

Related Articles feature; however, our implementation improves upon PubMed's interface because it allows users to compute this set of related documents for original queries in real time, while PubMed's Related Articles are pre-computed only for abstracts published in MEDLINE.

3.3.3 Calibration of the alignment-based similarity score Two sets of 'abstracts' were submitted to eTBLAST to evaluate its ability to quantitatively and statistically measure the performance of its two-step process including the alignment scoring algorithm, thus its performance as a similarity search engine. First, a set synthetic queries (abstracts) of random keywords was processed to measure the distribution of eTBLAST alignment-based scores to enable the computation of an expectation value. All abstracts within the MEDLINE06n baseline were analyzed to determine the size distribution of keywords (total words minus stop words). The frequency of all keywords in the entire database was measured as well. A set of 1000 pseudo-random queries with the same size distribution and word frequency distribution were synthesized using the built-in Perl pseudo-random number generator. The maximum observed score returned by each eTBLAST query was noted. Using the derived distribution (Synthetic random abstract scores, Fig. 5) we determined the average maximum observed score and the SD, 0.385 and 0.152, respectively. Inspection of the results from typical queries submitted to the eTBLAST server indicate that on average the top 200–300 returned abstracts are 2 SD above this random noise level (0.687). Second, we randomly selected 1000 abstracts from 2005 MEDLINE and submitted them to eTBLAST to quantitatively determine the eTBLAST alignment-based score for an identity comparison (Identity abstract scores, Fig. 5) and also for the most similar, non-identical abstract found (Most similar non-identity abstract scores, Fig. 5). Clearly, identical, similar and nonsensical comparisons can be distinguished.

4 DISCUSSION

We compared several TSS algorithms with one another utilizing both user testing and industry-standard evaluation tools. We showed that a cosine similarity function weighted with IDF and a low-base log function for term frequency produced the best results among similarity searches relying on word-vector strategies. We further showed that our novel sentence alignment algorithm offers an improvement over this baseline. This systematic evaluation has enabled us to define a set of preferences and select the best-performing approach for comparing an arbitrary text query against MEDLINE. Using the optimized two-step algorithm to process synthetic randomly constructed queries has enabled us to compute and output to the user a Z-score for each returned similar abstract. This optimized set of algorithms and associated parameters provides the basis for a new high-performance literature similarity search service, eTBLAST, requiring less effort on behalf of the searcher, acceptable computation time and a relatively conservative amount of infrastructure.

ACKNOWLEDGEMENTS

The authors wish to thank Natalie Prikhodko and Alexander Pertsemlidis for the creation of the prototype eTBLAST interface. The authors also wish to thank the Hudson Foundation for supporting this project. Support also came from the P.O'B. Montgomery Distinguished Chair.

Conflict of Interest: none declared.

REFERENCES

Altschul,S. *et al.* (1990) Basic local alignment search tool. *J. Mol. Biol.*, **215**, 403–410.

Bernstam,E. (2001) MedlineQBE (query-by-example). *Proc. AMIA Symp.*, 47–51.

Bradigan,P.S. and Mularski,C.A. (1989) End-user searching in a medical school curriculum: an evaluated modular approach. *Bull. Med. Libr. Assoc.*, **77**, 348–356.

Chen,J., Chau,R. and Yeh,C. (2004) Discovering parallel text from the world wide web. In *Proceedings of the second workshop on Australasian information security, Data Mining and Web Intelligence, and Software Internationalisation,* Dunedin, New Zealand, pp. 157–161.

Funk,M.E. and Reid,C.A. (1983) Indexing consistency in MEDLINE. *Bull. Med. Libr. Assoc.*, **71**, 176–183.

Granka,L., Joachims,T. and Gay,G. (2004) Eye-tracking analysis of user behavior in www search. In *ACM Conference on Research and Development in Information Retrieval (SIGIR),* University of Sheffield, UK.

Haynes,R.B. *et al.* (1990) Online access to MEDLINE in clinical settings. A study of use and usefulness. *Ann. Intern. Med.*, **112**, 78–84.

Hersh,W. (2003) *Information Retrieval: A Health and Biomedical Perspective*. 2nd edn. Springer-Verlag, New York.

Hersh,W. and Bhupatiraju,R. (2003a) TREC genomics track overview. In *Proceedings of the Text Retrieval Conference 2003,* Gaithersburg, MD, pp. 14–24.

Hersh,W. and Bhupatiraju,T. (2003b) TREC Genomics Track—Overview. In *Proceedings of the Text Retrieval Conference 2003,* Gaithersburg, MD, pp. 1–14.

Hersh,W. and Hickam,D. (1995) An evaluation of interactive boolean and natural language searching with an Online Medical Textbook. *J. Am. Soc. Inform. Sci.*, **46**, 478–489.

Joachims,T., Granka,L., Pan,B., Hembrooke,H. and Gay,G. (2005) Accurately interpreting clickthrough data as implicit feedback. In *ACM Conference on Research and Development in Information Retrieval (SIGIR),* Salvador, Brazil, pp. 154–161.

Jurafsky,D. and Martin,J. (2000) *Speech and Language Processing*. Prentice Hall, Uppers Saddle River, NJ.

Leonard,L. (1975) Inter-indexer consistency and retrieval effectiveness: measurement of relationships. PhD Thesis, University of Illinois, Champaign, IL.

McKibbon,K.A. and Walker-Dilks,C.J. (1995) The quality and impact of MEDLINE searches performed by end users. *Health Libr. Rev.*, **12**, 191–200.

Pertsemlidis,A. and Garner,H. (2004) Text comparison based on dynamic programming. *IEEE Eng. Biol. Med.*, **23**, 66–71.

Salton,G. (1983) *Introduction to Modern Information Retrieval*. McGraw-Hill, New York.

Salton,G. (1991) Developments in automatic text retrieval. *Science*, **253**, 974–980.
Van Rijsbergen,C.J. (1979) *Information Retrieval*. Butterworth, London.
Voorhees,E. (2003) Overview of TREC. In *Proceedings of the Text Retrieval Conference 2003,* Gaithersburg, MD, pp. 1–14.
Wiesman,F. *et al*. (1997) Information retrieval: an overview of system characteristics. *Int. J. Med. Inform*., **47**, 5–26.
Wildemuth,B.M. and Moore,M.E. (1995) End-user search behaviors and their relationship to search effectiveness. *Bull. Med. Libr. Assoc*., **83**, 294–304.
Yamamoto,E., Kishida,M., Takenami,Y., Takeda,Y. and Umemura,K. (2003) Dynamic programming matching for large scale information retrieval. In *Sixth International Workshop on Information Retrieval with Asian Languages,* Sapparo, Japan.