

A Relational Genomics Search Engine

Jay Urbain and Nazli Goharian

Abstract—We report on the development of a relational genomic search engine that integrates search of structured biological data and biomedical literature. After identifying an optimal preprocessing strategy, we present techniques for gene/protein term normalization, acronym expansion, and compound word normalization. Several retrieval models are evaluated on a large biomedical corpus using a common baseline. The system delivers results that meet or exceed the current state-of-the-art depending on the model used.

Index Terms—Biomedical computing, Database searching, Information retrieval, Medical information systems, Natural languages.

I. INTRODUCTION

Biomedical research involves the use and integration of a wide variety of structured and unstructured data, including descriptions of disease, drug interactions, protein descriptions, gene sequences, scientific literature, and meta-data.

As high-throughput techniques such as gene microarrays generate massive amounts of structured biological data, scientific literature plays an important role in the growth of biomedical research data and knowledge. The ability to integrate structured data with knowledge from scientific literature becomes a key component of experiments to identify new genes, diseases, and other biological processes that require further investigation. In addition, the literature itself can become a source of experiments as researchers turn to it to search for knowledge that drives new hypotheses and research [1, 10, 11, 12].

To meet the need for integrated search of structured data and biomedical literature, we report on the development of a scalable genomic retrieval engine developed using a standard relational database. Building a text retrieval engine on a

Jay Urbain (presenter) is with the Information Retrieval Laboratory, Illinois Institute of Technology Research Institute and the Computer Science Department, Illinois Institute of Technology, Chicago, IL USA (phone: 414-745-5102; fax: 215-902-5790; e-mail: urbain@iit.edu).

Nazli Goharian is with the Information Retrieval Laboratory, Illinois Institute of Technology Research Institute and the Computer Science Department, Illinois Institute of Technology, Chicago, IL USA (phone: 312-567-5704; fax: 312-567-5067; e-mail: goharian@iit.edu).

relational database allows simultaneous search of structured data from biological databases and text-based biomedical literature using a single SQL query. This facilitates query augmentation, enhanced indexing techniques, and efficient evaluation of retrieval strategies through modification of an SQL aggregate function. Additional queries can easily be developed for data and query analysis, and research efforts can focus on retrieval techniques rather than implementation details by leveraging off of the commercial database industry's investment in scalability, concurrency, and query optimization.

The system is evaluated on the TREC-2005 Genomics ad-hoc retrieval task which utilizes a corpus of 4,591,008 Medline citations (~15GB) and 50 query topics drawn from the information needs of molecular biology researchers [1].

The Medline Database is the premier bibliographic database for scientific articles in biomedicine consisting of more than 13 million citations. Each Medline citation includes article title, manually indexed medical subject headings (MeSH), and typically includes an abstract.

We evaluate the performance of several information retrieval models using standard SQL queries: BM25 [2], pivoted document length normalization (PDLN) [3] and language models with several smoothing and relevance strategies [4, 5].

Utilizing gene/protein term normalization, compound query terms, and Porter stemming, we received a Mean Average Precision (MAP) of 0.301 with BM25, 0.302 with a Jelinek-Mercer smoothed language model, and 0.308 with a relevance weighted language model. These results are among the top results for manual runs reported for the 2005 TREC ad-hoc Genomics track, demonstrating state-of-the-art results with a relational approach on commodity PC hardware.

II. RELATIONAL DATA MODEL

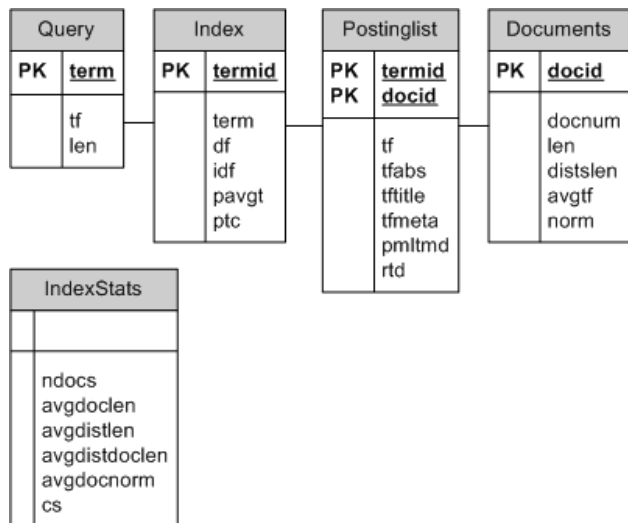
The mainstream approach in the development of information retrieval systems uses a customized inverted index to represent text with additional custom software required to integrate disparate structured and unstructured data sources.

Our genomic retrieval engine is based on a relational implementation of information retrieval functions [6] and uses relations to model an inverted index. Storing the full text in a relational database integrates the search of unstructured data with the traditional structured data search of a relational database management system (RDBMS).

As shown in **Figure 1**, the inverted index is implemented as

a set of relational database tables: *index*, *postinglist*, and *documents*. In addition, an *indexstats* table is created for capturing corpus wide statistics to support language models and various normalization schemes. A query table is created for storing the current topic.

Figure 1: Relational Model



All retrieval functions are implemented using standard SQL.

The following query implements the cosine retrieval function with pivoted document length normalization:

```

select p.docid, max(d.docnum) docnum,
       sum(i.idf*(1+ln(q.tf))*idf*(1+ln(p.tf))*d.NORM ) as sc
from index i, postinglist p, documents d, query q
where p.docid=d.docid
and i.termid=p.termid
and i.term=q.term
group by p.docid
order by sc desc;
  
```

where $d.NORM = 1/(0.75+((0.25/avgdoclen)*len))$;

By modifying the aggregate SUM function, additional retrieval functions can be implemented. The following implements the popular probabilistic BM25 model:

```

select p.docid, max(d.docnum) docnum,
       sum( ln((s.ndocs-i.df+0.5)/(i.df+0.5))*
            (((k1+1)*p.tf)/(k1*((1-b)+b*(d.len/s.avgdoclen))+p.tf))
            ((k3+1)*q.tf/(k3+q.tf)) ) as sc " +
from index i, postinglist p, documents d, query q
where p.docid=d.docid
and i.termid=p.termid
and i.term=q.term
  
```

```

group by p.docid
order by sc desc;
  
```

And the following implements a relevance-weighted language model:

```

select p.docid, max(d.docnum) docnum,
       sum(ln(1+((1-lambda)/lambda)*
            (p.tf/d.len)*(s.cs/i.df)*(docPrior) ) ) sc
from index i,postinglist p,documents d, query q, indexstats s
where p.docid=d.docid
and i.termid=p.termid
and i.term=q.term
group by p.docid
order by sc desc;
  
```

Different term weighting schemes can be accommodated for title, abstract, and MeSH terms by modifying the aggregation equation as follows:

```

select p.docid, max(d.docnum) docnum,
       sum(i.idf*(1+ln(q.tf))*idf*
            (w1*(1+ln(p.tftitle))+
            (w2*(1+ln(p.tfabs))+
            (w3*(1+ln(p.tfmesh))))
            *d.NORM ) as sc ...
  
```

By loading TREC (Text REtrieval Conference) evaluation files containing listings of relevant and non-relevant documents into a database table, SQL reports can be generated to evaluate queries, relevant documents, and documents retrieved.

III. PREPROCESSING

A. Frequent/infrequent terms

To improve performance, frequent and infrequent terms were pruned from the index. Additional stop words were identified that either occur frequently in genomics literature (“disease”, “gene”, “biological”, etc), or do not occur frequently, but do not support relevance (“study”, “process”, “analysis”, “information”, etc). In our evaluation, removal of non-relevant terms improved MAP between 0.5 and 2.0 points depending on the retrieval model used.

B. Stemming

Porter [9] postfix stemming was utilized (on non-acronym terms) to reduce term variations. Our evaluation showed postfix stemming outperformed combined pre- and post-fix stemming by approximately 0.5 point of MAP, and use of no stemming by approximately 2.5 points.

C. Gene/protein term normalization

Gene/protein normalization [7] requires identification of terms with mixed case, alpha-to-numeric, or numeric-to-alpha character transitions that are not separated, separated by a space, or separated by a hyphen. Sample terms include Nurr77, ApoE, NM23, and TGF-beta1. Nurr77, Nurr-77, and Nurr 77 would all be normalized to “Nurr77”.

To capture additional variations, each component variation of a normalized term is generated. For example, TGF-beta1 is first normalized to “tgfbeta1”, and then each component of the normalized term is generated prior to removing stop words: “tgfbeta”, “beta1”, “tgf”, “beta”, and “1”.

D. Compound Words

Numerous variations in the use of single and compound words are prevalent in biomedical literature. For example, “immuno precipitant” is frequently referred to as “immunoprecipitant”, and “glutathione S-transferase” is often referred to as “glutathionestransferase”. We attempted to mitigate these variations by generating compound words by conflating successive non-stop-word query terms (bi-grams), and augmenting the query with these compound words if they were prevalent in the index.

E. Acronym expansion/generation

During indexing, adjacent acronyms and their expansions, where either the acronym or the expansion was found in parenthesis, were parsed and stored along with their frequency of occurrence. Queries can then be expanded either with an expansion of an acronym present in a query, or with an acronym provided the acronym’s expansion terms are present in the query. For example, the acronym “ApoE” can be expanded with “apolipoprotein e”. Alternatively, the acronym “IP” can be added when the terms “immuno precipitant” are present in the query.

IV. QUERY PROCESSING

We evaluated each of the following retrieval functions:

- Pivoted Document Length Normalization (PDLN) [3].
- BM25 [2].
- Language models using Jelinek-Mercer (JM), Dirichlet (D), and Absolute Discounting (AD) [4], and a model based on a Relevance Weighting (RW) scheme [5].

The same preprocessing, i.e., term normalization, stop word removal, and Porter stemming utilized for indexing was utilized for query processing. Each of the following retrieval models were executed on the same index with the same preprocessing.

A. Cosine with pivoted document length normalization

The standard cosine retrieval function normalizes document similarity measurements by document length.

Cosine:

$$\sum_{wq} \frac{idf * \ln(1 + tf_q) * idf * \ln(1 + tf_d)}{docLen}$$

$idf * \ln(1 + tf_q)$ represents the weight assigned to each query term. idf is a standard information retrieval term weight for identifying how well any given term discriminates between documents and is equal to $\ln(N/df)$. N represents the number of documents in the collection and df is the number of documents the term occurs in. tf_q represents the frequency of occurrence of any given query term. The logarithm of term frequency is used since raw term frequencies tend to weigh multiply occurring terms too heavily.

Similarly, $idf * \ln(1 + tf_d)$ represents the weight assigned to each document term.

Standard cosine document length normalization techniques have been shown to over penalize longer documents, i.e., shorter documents are more likely to be retrieved and less likely to be relevant [3]. Utilizing a slope “ s ” adjustment, pivoted document length normalization (PDLN) adjusts the document normalization about a pivot (usually the average document length) such that the retrieval curve more closely represents the likelihood of retrieval. We therefore utilize cosine with PDLN in our evaluation.

PDLN:

$$\sum_{wq} \frac{idf * \ln(1 + tf_q) * idf * \ln(1 + tf_d)}{(1 - s) + s * (\frac{doclen}{avgdoclen})}$$

We received our best results with $s=0.25$

B. Probabilistic

We utilized the standard BM25 probabilistic algorithm [2].

BM25:

$$\sum_{wq} \ln \left(\frac{N - df + 0.5}{df + 0.5} \right) \left(\frac{(k_1 + 1) * tf_d}{k_1 * (1 - b) + b * (\frac{docLen}{avgDocLen}) + tf_d} \right) \left(\frac{(k_3 + 1) * tf_q}{k_3 + tf_q} \right)$$

$\left(\frac{N - df + 0.5}{df + 0.5} \right)$ represents term relevance and is similar to idf .

$$\left(\frac{(k_1+1)*tf_d}{k_1*(1-b)+b*\left(\frac{docLen}{avgDocLen}\right)+tf_d} \right)$$

provides adjustable document term weighting using constant k_1 , and adjustable document length normalization with constant b .

$$\left(\frac{(k_3+1)*tf_q}{k_3+tf_q} \right)$$

provides adjustable query term weighting with constant k_3 .
We received our best results with $k_1=1.4$, $k_2=0$, $k_3=7$, and $b=0.75$.

C. Language Models

We evaluated a relevance weighted unigram language model with and without document priors, and several variations of most-likelihood-term unigram language models. The most-likelihood-term models were evaluated with Jelinek-Mercer, absolute discounting, and Dirichlet smoothing using uniform document priors. Traditional and KL-Divergence [8] formulations were evaluated.

To improve performance for the traditional implementation, the *where* clause of the SQL query was modified to only include counts for documents which included at least one term that matched a query term.

The KL-Divergence formulations and the relevance-weighted models do not require statistics from unmatched query terms and are therefore much more efficient.

1) Jelinek-Mercer

Jelinek-Mercer smoothing utilizes a linear interpolation to distribute the probability mass between terms *seen* in the document with the likelihood of the term occurring in the collection.

LM-JM:

$$\sum_{wq} \ln((1-\lambda)*P_{mi}(w|d) + \lambda * P(w|C))$$

$P_{mi}(w|d) = tf_d / docLen$ represents the most-likelihood probability of a term given a document.

$P(w|C)$ represents the frequency of the term in the collection.

We received our best results with $\lambda = 0.1$

2) Bayesian Smoothing with Dirichlet Prior

Bayesian smoothing with Dirichlet Prior uses the Dirichlet distribution of terms as the prior for Bayesian analysis.

LM-D:

$$\sum_{wq} \ln\left(\frac{tf_d + \mu * P(w|C)}{docLen + \mu}\right)$$

$\mu * P(w|C)$ represents the Dirichlet distribution of query terms.

We received our best results with $\mu = 2000$.

3) Absolute Discounting

Absolute discounting is similar to Jelinek-Mercer, but instead of lowering the probability of seen words by multiplying it by $(1-\lambda)$, it subtracts a constant δ .

LM-AD:

$$\sum_{wq} \ln\left(\frac{\max(tf_d - \delta, 0)}{docLen} + \frac{\delta * distDocLen}{docLen}\right)$$

The second term, $\frac{\delta * distDocLen}{docLen}$, ensures all probabilities sum to one.

We received our best results with a discount constant $\delta = 0.8$

4) Relevance Weighted

The relevance weighted model interprets retrieval language models as incorporating the odds of probability of relevance as used in the traditional probabilistic model of information retrieval, and also as members of the family of $tf*idf$ weighted algorithms developed for the vector space model [5]. The relevance weighted model is also very attractive in only requiring matching terms in its computation.

LM-RW:

$$\sum_{wq} \ln\left(\left(\frac{1-\lambda}{\lambda}\right) * \left(\frac{tf_d}{dft}\right) * \left(\frac{cs}{docLen}\right)\right) + \ln\left(\frac{docLen}{cs}\right)$$

$\left(\frac{1-\lambda}{\lambda}\right)$ represents the (inverse) odds probability of relevance.

$\left(\frac{tf_d}{dft}\right)$ represents the $tf*idf$ weight based on the term

frequency of the term within the document (tf_d), and the document frequency of the term within the collection (dft).

$\left(\frac{cs}{docLen}\right)$ cs represents the collection frequency, and $\left(\frac{1}{docLen}\right)$ represents the inverse length of the document.

$\ln(docLen / cs)$ is the query-independent document prior represented as the probability of any given term occurring within a given document.

We received our best results with $\lambda = 0.1$

V. SYSTEM DESCRIPTION

The indexing and retrieval engine was developed in Java using the Oracle 9i Standard Edition database. The system is platform and database independent. All evaluations were performed on a 3.1GHz Pentium 4 PC with 4 GB of memory.

VI. RESULTS

In addition to model dependent parameters, we experimented with stemming, term normalization, and term variations. We later present our comparative findings using the best parameters for each given model. First, as an illustrative example, in **Table 1**, we present results for the PDLN baseline, where $s=0.25$, the baseline with stemming, the baseline with stemming and gene/protein term normalization, and finally, also including term variants.

Table 1: Preprocessing Evaluation

Preprocessing	MAP	% imp.
Baseline PDLN	0.204	
stem	0.213	4.4%
stem, term norm.	0.251	23.0%
stem, term norm. w/ variants	0.266	29.4%

In **Table 2**, we summarize our results for all models considered. For each model, only the top MAP is presented. Baseline scores for the percentage improvement assume PDLN based retrieval. As shown, our best results were achieved using LM-RW with the BM25 and LM-JM MAP scores being virtually identical to that of LM-RW. BM25 was more stable for a wide range of parameters, and individual queries executed in the 0.5 to 2 seconds range. This execution speed is approximately ten-fold faster than LM-RW and the KL-Divergence formulations for LM-JM, LM-D, and LM-AD.

Table 2: Retrieval Strategy Evaluation

Retrieval	Parameters	MAP	% imp.
PDLN	$s=0.25$	0.266	-
BM25	$k1=1.4, k3=7, b=0.7$	0.301	13.2%
LM-JM	$\lambda = 0.1$	0.302	13.5%
LM-D	$\mu = 2000$	0.255	-4.1%
LM-AD	$\delta = 0.8$	0.292	9.8%
LM-RW	$\lambda = 0.1$	0.308	15.8%

Use of the KL-Divergence formulation for LM-JM, LM-D, and LM-AD improved execution speed but reduced precision in all cases, so results are reported using the standard formulation. The language models appear more sensitive to document length variations (~25% of citations in the collection have no abstract) and stop word selection, both of which directly impact collection statistics. Addition of genomics collection specific stop words improved the performance of BM25 by approximately 5%, however the addition of the same stop terms improved the performance of LM-RW, LM-AD, and LM-JM by approximately 25% demonstrating the sensitivity of the language models to the collection statistics. Further evaluation of pivot techniques to refine document length normalization strategies for the models did not significantly improve performance.

We evaluated a number of query expansion techniques. Positive Rochio pseudo relevance feedback (RF) was evaluated for several variations in the number of top docs, number of terms, and number of iterations used, but did not improve MAP in any case. To limit the selection of non-relevant RF terms, we evaluated selection of RF terms from the largest result set clusters (hierarchical agglomerative) for different numbers of top docs and at different levels of the clustering hierarchy. Result set clustering improved RF performance, but did not improve performance over using no RF at all (0.3010 with RF/clustering vs. 0.3013 without).

Numerous variations in the use of compound words are prevalent in biomedical literature. “nucleoside diphosphate” is often referred to as “nucleosidediphosphate”, and “auto immune” is often referred to as “autoimmune”. We explored several query expansion strategies using bigrams generated from consecutive non-stop-word terms including: forming compounds with and without postfix stemming of the first term, reversing the order of the terms (“tagged protein” becomes “proteintag”), and conjugating the components of gene/protein names with surrounding text. Compounds were only added to the query if they already existed in the lexicon above a document frequency threshold. Except for generating compounds as part of our acronym and gene/protein term normalization schemes, none of the compound word generation techniques we evaluated improved average precision. The best technique was the simplest: join sequential non-stop-word terms without stemming, and then stem the final compound.

Many of the documents retrieved with our compound word generation technique were not in the original pool of documents marked relevant and nonrelevant from the collection evaluation (“qrels”) file. After further review of these documents, many of the documents appear relevant to their respective queries, so we believe this technique does have potential.

Finally, we evaluated acronym expansion techniques. During indexing, adjacent acronyms and their expansions, where either the acronym or the expansion was found in parenthesis, were parsed and stored along with their frequency of occurrence. Query acronyms were then expanded with terms from their most frequently occurring expansion. This technique significantly improved performance for some queries, but hurt performance for other queries due to either increased generalization or use of the incorrect expansion in the query. In one case, the acronym “IP” was expanded to “ischemic precondition”, however the correct expansion was “immunoprecipitant”. Selection of the correct (lower frequency) expansion would have improved performance. Conflating acronym expansions already defined in queries into compound terms increased MAP by approximately 0.5%.

VII. CONCLUSION

We developed a rapid-prototype, genomic-literature, retrieval engine using conventional relational technology. In doing so, we captured the ability to integrate structured components into our search.

Using this engine, we evaluated multiple retrieval models. Our initial findings demonstrate the statistical equivalence of the BM25 and relevance weighting language models in terms of accuracy. However, in our implementation, the BM25 model executed significantly faster.

Future efforts include enhancing queries and indexing through incorporation of data from external databases, and improved techniques for biomedical term stemming, acronym expansion and for matching the numerous variations of compound words prevalent in biomedical literature.

REFERENCES

[1] W. Hersh, et al. TREC 2005 Genomics track overview. Proceedings of the Fourteenth Text REtrieval Conference, Gaithersburg, MD.

[2] S.E. Robertson, S. Walker (2000). Okapi/Keenbow at TREC-8. NIST Special Publication 500-246.

[3] A. Singhal, C. Buckley, and M. Mitra (1996). Pivoted Document Length Normalization. 19th ACM SIGIR.

[4] C. Zhai and J. Lafferty (2001). A study of smoothing methods for language models applied to ad hoc information retrieval. 24th ACM SIGIR.

[5] D. Hiemstra and A. P. de Vries (2000). Relating the new language models of information retrieval to the traditional retrieval models. Technical Report TR-CTIT-00-09, Centre for Telematics and Information Technology.

[6] D. Grossman, O. Frieder, D. Holmes, and D. Roberts (1997). Integrating Structured Data and Text: A Relational Approach. JASIS, 48(2).

[7] S. Buttcher, C. Clarke, and G. Cormack (2004). Domain-specific synonym expansion and validation for biomedical information retrieval TREC 2004 Genomics Track experiments at Patolis. Proceedings of the Thirteenth Text REtrieval Conference, Gaithersburg, MD.

[8] C. Zhai and J. Lafferty (2001). Model-based feedback in the KL-divergence retrieval model. Proceedings of the Tenth International Conference on Information and Knowledge Management (CIKM 2001), pages 403–410.

[9] M.F. Porter (1980). An algorithm for suffix stripping. Program, 14:130–137.

[10] S.B. Davidson, C. Overton, and P. Buneman (1995). Challenges in integrating biological data sources. Journal of Computational Biology, 2(4), 557-572.

[11] W. David Fenstermacher (2005). Introduction to Bioinformatics. Journal of the American Society for Information Science and Technology, Volume 56, Issue 5, Pages 440 – 446.

[12] W. John MacMullen, and Sheila O Denn. (2005). Information problems in molecular biology and bioinformatics. Journal of the American Society for Information Science & Technology 56(5), 447-456.