TREC 2003 Genomics Track Experiments at UTA: Query Expansion with Predefined High Frequency Terms

Ari Pirkola and Erkka Leppänen
University of Tampere (UTA), Finland
Department of Information Studies
pirkola@cc.jyu.fi

Abstract We studied the effects of query expansion and query structure on retrieval performance. Two sets of words frequent in relevant documents for Genomics Track's training topics were collected, the first manually and the second automatically. The high frequency words collected and the names of organisms designated in the test topics, were used as expansion keys in gene name queries formed from the final test topics. The results indicated that Boolean structured queries expanded with automatically collected high frequency words and names of organisms performed considerably better than queries containing gene names only as keys. In the Boolean queries the expansion keys were categorized based on the aspects they represent in the documents discussing gene function. All the structured queries performed better than unstructured queries where each key contributed equally to document weights. In the structured queries gene names were assigned more weight than the expansion keys.

1. Introduction

TREC Genomics Track's primary task was defined as follows: *For gene X, find all Medline references that focus on the basic biology of the gene or its protein products from the designated organism. Basic biology includes isolation, structure, genetics and function of genes/proteins in normal and disease states.* Documents for which there were GeneRIFs were considered relevant. As the focus of GeneRIFs is to provide information on the functions of genes (and proteins), basically documents discussing gene function were searched for in the primary task.

In this research we studied, first, which types of words other than gene names are useful in searching for documents for the given task and, second, how queries using these words should be formulated for the best possible retrieval performance (the question of query structure). For the first research problem we identified two sets of words (manual and automatic identification) whose frequencies were higher in documents relevant for Genomics Track's *training topics* than in the rest of the documents in the Medline test collection. The power of these words to actually discriminate between gene function and other documents was evaluated in experiments where the words were used as expansion keys in the final test queries. The performance of the expanded queries was compared to the performance of baseline queries containing gene (and protein) names only as query keys. For the runs that were submitted to NIST the expansion keys were selected manually. Most of the runs we did were additional runs, and for these the expansion keys were selected automatically. In the additional tests names of organisms were also used as expansion keys.

The second research problem considered the effects of query structure on retrieval performance. The test system of this study was the *InQuery* retrieval system, a probabilistic retrieval system. In InQuery queries can be structured using a variety of query operators. The assumption was that for good performance gene names should be weighted more than other keys, and that in addition to gene names other aspects of relevant documents should also be included in queries.

These factors were taken into account in the *Boolean structured queries* tested in the study. The results showed that performance was improved considerably when gene name queries were expanded with names of organisms. The best queries, however, were extensively expanded Boolean structured queries where gene names were weighted structurally more than the other keys, and where expansion keys were categorized based on the aspects they represent in gene function documents. The categories were: (1) names of organisms (given in the test topics), (2) high frequency special terms related to genes and proteins, and (3) high frequency general biological terms. Expansion key categories 2 and 3 were selected automatically from documents relevant for Genomics Track's training topics. The Boolean structured queries performed considerably better than queries containing gene names only as keys (baseline) and unstructured queries. In the unstructured queries gene names and expansion keys were weighted equally.

2. Methods and data

2.1 Indexing and query keys

We used the following approaches and techniques in indexing and for query keys:

- Genomics Track's test collection for the primary task was a subset of the Medline collection. Each record consisted of several fields. We indexed the text fields TI (title), AB (abstract), and MH (MeSH headings). (And PMID, PubMed Identifier.)
- Letters were normalized to lower case.
- Query keys and the words of documents were normalized using the morphological analyzer *Kstem*, which is part of InQuery.
- Only letters (a-z) and numbers (0-9) were indexed. Hyphens, slashes and other characters in strings than letters and numbers were replaced by spaces, and were not searchable.
- Strings containing both letters and numbers were decomposed into separate alphabetical and numerical strings. For example, the string *AW986256* was converted into *aw 986256*. In searching the decomposed strings were combined by means of a proximity operator of #od4. For the string *AW986256* the proximity statement was as *#od4(aw 986256)*.

2.2 Retrieval system and query operators

The test system was the *InQuery* retrieval system (Allan et al., 2000; Broglio et al., 1994). InQuery is a probabilistic retrieval system based on the Bayesian inference net model. Queries can be presented as a bag of word queries, or they can be structured using a variety of query operators. In this study we considered structured queries. For the structured queries high frequency words were selected as expansion keys (Section 2.3). The expansion keys were grouped into categories (subqueries) based on the aspects they represent in documents. In these types of queries different aspects of documents discussing the functions of genes contribute to retrieval. Query operators were applied in such a way that more weight was assigned to gene names than to other keys.

Boolean conjunction ("and") formed the basis of queries. In InQuery Boolean conjunction is denoted using the operators of #band or #filreq (and for phrases #odn, see below). All the argument keys of the *#band-operator* must occur in a document in order for the operator to contribute to the weight computed for that document. Otherwise #band contributes to the document score like the #and-operator. For the #band-operator, see Pirkola and Järvelin (2001). The weight of the *#and-operator* is computed as the product of the weights of its arguments. The syntax of *#filreq* (filter require) operator is #filreq(arg1 arg2). Documents for the first argument are returned and ranked if and only if the second argument would return documents.

Other InQuery operators used in the queries of this study were #sum, #syn, #odn, and #phrase. For the *#sum-operator*, the system computes an average weight of query keys (subqueries). The *#syn-operator* treats its operand query keys as instances of the same query key. The benefits of combining keys by the #syn-operator is discussed in Darwish and Oard (2003). Phrases were searched for using the proximity operators of #odn and #phrase. The *#odn*-operator is a Boolean conjunction operator. The operator retrieves documents where all the arguments of the operator appear in a specified order. The window size *n* refers to the number of spaces between words in the text. In the queries of this study n=4 was used as a window size. The *#phrase*-operator is treated as #odn-operator, if the terms within the phrase-operator occur together in a collection. If the terms do not co-occur in the collection the phrase operator is turned into a #sum-operator.

2.3 Selecting expansion keys

We tested in this study whether words typical in gene function documents are helpful for the given retrieval task when used as expansion keys in gene name queries. Two lists of expansion keys were collected, the first manually and the second automatically. The basic assumption behind this approach was that those words whose frequencies were high in relevant documents for the Genomics Track training topics with respect to their frequencies in other Medline documents are good discriminators and good keys in queries, designating generally documents discussing gene function.

In manual word selection a list of words was collected that in an intellectual analysis seemed to be frequent in the documents relevant for the training topics. The words were divided into two categories: (a) *gene function terms*, and (b) other high frequency terms, for example, *cell*, *mrna*, and *sequence*. The latter words are called *general terms* in Section 3.1. The two categories formed two different subqueries in the structured queries.

The automatic expansion key selection was done as follows. For each training topic at most two relevant documents were chosen for statistical analysis. The total number of 91 relevant documents and approximately 24 000 words were analysed. The same number of randomly selected documents (approximately the same number of words) was analysed. In both cases words were normalized using the Kstem morphological analyser, and stopwords were removed. For each remaining word the frequency of a word was computed. Those words whose frequencies were low (freq. < 10) were removed.

A word i was chosen as an expansion key based on the following criteria:

(1) Its term frequency in the relevant documents ($tf_{i, rel}$) was higher than a given threshold; the threshold was $tf_{i, rel} = 30$ (in other words, word's term frequency on the average was higher than 30/91).

(2) Its relative frequency was higher than a given threshold. The relative frequency of word i is its frequency in the relevant documents ($tf_{i, rel}$) divided by its frequency in the random documents ($tf_{i, ran}$). Three thresholds were tested:

$tf_{i, rel} / tf_{i, ran} = 3.1$	(No of expansion keys = 32)
$tf_{i, rel} / tf_{i, ran} = 3.7$	(No of expansion keys = 20)
$tf_{i, rel} / tf_{i, ran} = 4.6$	(No of expansion keys = 11)

The numbers in parentheses show the number of words kept for expansion as the threshold was applied.

The automatically selected word list with the threshold 3.1 is presented in the Appendix.

In both cases (manual and automatic selection of expansion keys) the selected words were added to gene name queries (baseline) to yield expanded queries. The performance of the expanded queries was compared to that of baseline queries. Manually collected expansion keys were used in the runs submitted to NIST while the automatically selected words were used in the additional runs.

3. Experiments

In summary, we tested in this study the effects of (1) manually and (2) automatically selected high frequency words as expansion keys, and (3) different query structures. Different types of expansion keys were put into different subqueries in queries. In the experiments 2 and 3 we also tested the effects of organism name keys. Next, the query types used in the three experiments are presented.

3.1 Manually selected expansion keys

Baseline

The baseline queries were formulated on the basis of Genomics Track's test topics, and they contained all the gene and protein names occurring in the topics.

Phrases were searched for using the proximity operator of #od4, e.g., *#od4(hematopoietic growth factor)*. As there is much variation how phrasal expressions are written in texts, phrase components were also repeated as single keys in queries, and for phrases consisting of more than 3 phrase components several proximity statements were constructed, with consecutive phrase components appearing in 1-3 proximity statements, as in the example below:

```
#od4(camp responsive element) #od4(responsive element binding) #od4(element binding protein)
#od4(binding protein 2)
```

An example of a baseline query is presented below (query 31):

```
#sum(gnrhr #od4(gonadotropin releasing hormone) #od4(releasing hormone receptor) gnrhr gonadotropin releasing
hormone receptor)
```

Test queries (runs submitted to NIST)

In the training experiments we tested various combinations of query operators. The most effective query types turned out to be those based on InQuery's *band*- and *filreq*-operators. They were named *utaband* and *utafil*. In the *utaband* queries in addition to the #od4-operator the #phrase-operator was used for phrases. The utaband queries contained 74 manually selected expansion keys, and the utafil queries 58 expansion keys. The structure of *utaband* is schematically as follows:

```
#band(
#and(#sum(gene names) #syn( gene function terms))
#and (#sum(gene names) #syn(general terms))
)
```

The structure of *utafil* is schematically as follows:

```
#filreq(
#sum(gene names #syn(gene function terms) #syn(general  terms))
#syn(gene names [as single keys])
)
```

The last #syn-subquery functions as a filter, basically being a Boolean conjunction restriction.

3.2 Automatically selected expansion keys, and organism names as expansion keys

Baseline

The baseline was the same as in the experiments presented in Section 3.1.

Test queries

The expansion key categories were as follows:

1. Organism names (*on*) designated in the Genomics Track's test topics. However, instead of using the Latin names used in the test topics, we used the MeSH terms *human*, *drosophila*, *rats* and *mice*.

2. Automatically selected high frequency terms related to genes and proteins. These are called *special terms* below.

3. Automatically selected high frequency general biological terms, i.e., other terms than those in the cases 1 and 2 frequent in documents discussing the functions of genes. These are called *general terms* below.

Obviously, the boundary between the categories 2 and 3 is diffuse, in some cases causing classification difficulties.

All the special and general terms collected are shown in the example queries below and in the Appendix.

In queries the terms of the category 2, as well as the terms of the the category 3, were combined by the #syn-operator to yield a #syn-subquery. The subqueries were combined with each other as follows (*gn* refers to gene names):

- gn (baseline)
- gn + on (see Section 3.3)
- gn + general terms
- gn + special terms
- gn + on + general terms
- gn + on + special terms
- gn + on + general + special terms

Query 31 with 32 expansion keys of the type *gn + on + general + special* below provides an example of the queries presented in this section:

#band(#sum(gnrhr #od4(gonadotropin releasing hormone) #od4(releasing hormone receptor) gnrhr gonadotropin releasing hormone receptor) rats #syn(cytology genetic membrane cell metabolism region site level receptor) #syn(amino alpha beta kinase tyrosine protein gene mutant mutation sequence activate expressing activation apoptosis binding expression regulate transcription induce pathway signal domain function))

The first #syn-subquery includes the general terms while the second #syn-subquery includes the special terms.

We call the query structure above *#band/#syn structure*, and it was used in all queries presented in this section, only the expansion keys were different in different types of queries. As can be seen in the example, in the #band/#syn structure the expansion keys of a certain category are combined with the #syn-operator.

The gene names are combined with the #sum-operator. The #sum-subquery, the #syn-subqueries, and organism name are combined with the #band-operator.

3.3 Query structures

In the query structure experiments, we studied #band/#syn structured (presented in Section 3.2), #and/#band structured, and unstructured queries. In this section we present the last two query types. The #and/#band structure gave good results in the training tests, and was chosen for the final tests.

The #and/#band structure is schematically as follows:

```
#and(
#band(#sum(gene names) #syn(general terms))
#band(#sum(gene names) #syn(special terms))
#band(#sum(gene names) organism name)
)
```

The first example below represents a #and/#band structured query, and the second one an unstructured query (query 31 with 32 expansion keys).

#and/#band structured query

```
#and(#band(#sum(gnrhr #od4(gonadotropin releasing hormone) #od4(releasing hormone receptor) gnrhr gonadotropin
releasing hormone receptor) #syn(cytology genetic membrane cell metabolism region site level receptor))
#band(#sum(gnrhr #od4(gonadotropin releasing hormone) #od4(releasing hormone receptor) gnrhr gonadotropin
releasing hormone receptor) #syn(amino alpha beta kinase tyrosine protein gene mutant mutation sequence activate
expressing activation apoptosis binding expression regulate transcription induce pathway signal domain function))
#band(#sum(gnrhr #od4(gonadotropin releasing hormone) #od4(releasing hormone receptor) gnrhr gonadotropin
releasing hormone receptor) rats))
```

Unstructured query

```
#sum(gnrhr #od4(gonadotropin releasing hormone) #od4(releasing hormone receptor) gnrhr gonadotropin releasing
hormone receptor cytology genetic membrane cell metabolism region site level receptor amino alpha beta kinase
tyrosine protein gene mutant mutation sequence activate expressing activation apoptosis binding expression regulate
transcription induce pathway signal domain function rats)
```

Moreover, in addition to gn+on/ #band queries (Section 3.2) we tested *gn+on* queries where the gn- and on-subqueries were combined with the #and-operator. Examples of both query types are presented below.

Gn+on / #band

```
#band(#sum(gnrhr #od4(gonadotropin releasing hormone) #od4(releasing hormone receptor) gnrhr gonadotropin
releasing hormone receptor) rats)
```

Gn+on / #and

```
#and(#sum(gnrhr #od4(gonadotropin releasing hormone) #od4(releasing hormone receptor) gnrhr gonadotropin
releasing hormone receptor) rats)
```

4. Results

The results of the expansion experiments are presented in Table 1 (manual selection of expansion keys) and Table 2 (automatic selection of expansion keys, and organism names as expansion keys). Table 2 also presents the results of query structure experiments.

Table 1. Retrieval performance of queries with manually selected expansion keys.

Query type	Average precision	% change
Baseline (gn)	0.1896	-
Utaband	0.1927	+1.6
Utafil	0.1931	+1.8

Table 2. Retrieval performance of queries with automatically selected expansion keys, and organism names as expansion keys; the effects of query structure.

Query type	Average precision	% change
Baseline (gn)	0.1896	-
Gn+on / #band	<u>0.2600</u>	<u>+37.1</u>
Gn+on / #and	0.2278	+20.1
Gn+general / #band/#syn	0.2042	+7.7
Gn+ special / #band/#syn	0.1999	+5.4
Gn+on+general / #band/#syn	0.2693	+42.0
Gn+on+special / #band/#syn	0.2606	+37.4
Gn+on+general+special / #band/#syn	0.2648	+39.7
Gn+on+general+special / #and/#band	<u>0.3097</u>	<u>+63.3</u>
Gn+on+general+special / unstructured	0.1692	-10.8

As shown in Table 1 there is only a small difference between the performance of baseline and the performance of queries containing manually selected expansion keys. The difference between *utaband* and *utafil* is also small. It should be noted that the utaband and utafil queries did not contain organism name keys.

The run *gn+on* (organism names as expansion keys) where the two subqueries (gn and on) are combined with the #band-operator performs considerably better than baseline (Table 2). For *gn+on* queries average precision is 0.2600. For the baseline the corresponding number is 0.1896. The performance of *gn+on* is lower in the case of the #and-operator but is still better than baseline's performance.

Table 2 shows the results for the queries where the expansion keys were selected on the basis of the threshold of 3.1 (Section 2.3) as they gave better results than the queries with the thresholds of 3.7 and 4.6.

The use of *special* and *general* terms as expansion keys is useful: the *gn+on+ general+special* / #and/#band queries give the best results among all query types tested in this study (average precision 0.3097). It should be noted that the structure of the *gn+on+general+special* / #and/#band queries and the *gn+on* / #band queries (average precision 0.2600) is the same. In both a gene name subquery is combined to an organism name subquery with the #band-operator. The difference is that the *gn+on* / #band queries do not contain the automatically selected expansion keys.

The unstructured queries containing all the expansion keys perform poorly (Table 2). For them average precision is 0.1692, while the average precision for the #band/#syn and #and/#band queries containing all expansion keys is 0.2648 and 0.3097, respectively.

5. Discussion and conclusions

In this study we explored an approach to concept analysis similar to that used in traditional Boolean searching. Based on the concept analysis the concepts of a request which represent different aspects of the request (conjunctive or restrictive concepts) are combined with the and-operator, and those concepts which represent the same aspect (disjunctive concepts) are combined with the *or*-operator with one another. For experienced users, Boolean and natural language searching have been shown to yield comparable results (Hersh et al., 1998). We analysed documents instead of requests, and formulated Boolean types of queries where terms representing the same aspect were put into the same category (subquery) in a query. The test system of this study was the *InQuery* retrieval system, a probabilistic retrieval system. Probabilistic query operators corresponding to the Boolean *and* and *or* operators were used in the queries. We found that retrieval performance improved remarkably with respect to gene name queries by expanding queries with names of organisms and automatically selected terms whose frequencies were high in gene function documents, and by using Boolean #and/#band structured queries. Extensively expanded queries gave the best results. The benefits of the #and/#band structure are that gene names are weighted more than other keys and that the #and/#band structure captures in a balanced way the different aspects involved in gene function documents. Different documents discussing gene function share the same aspects and the same terms. We identified four aspects (gene names and three aspects for expansion), but this is a crude categorization which should be elaborated and analysed more carefully in terms of biological processes and structures. The results, however, show that it is possible to improve results by using predefined high frequency discriminating words as query keys.

The most effective expansion key type was organism name. The explanation for this seems obvious. Different organisms contain the same genes which are named similarly. Unless organism name is given in a query, documents discussing the gene of interest in another organism may be retrieved.

There are many factors affecting the effectiveness of the expanded queries. Their performance depends in particular on which individual keys are used, the number of keys used, and the way the keys are applied in queries. In this study the latter two questions were investigated. The first is a question for future research. Also, based on these initial experiments on the use of predefined discriminating words as expansion keys, a more sophisticated scheme for the identification of discriminating words might be developed in future research. In addition to the relative term frequency that we used as a basis of the selection of expansion keys, the use of document frequency, for example, in such a scheme might be useful.

Acknowledgements

The Inquery search engine was provided by the Center for Intelligent Information Retrieval at the University of Massachusetts.

This research was supported by the Academy of Finland (funding decision 80142).

References

Allan, J., Connell, M.E., Croft, W.B., Feng, F.-F, Fisher, D. and Li, X. 2000. Inquery and TREC-9. *The Ninth Text REtrieval Conference (TREC-9),* Gaithesburg, MD. Available at: http://trec.nist.gov/pubs/trec9/t9_proceedings.html

Broglio, J., Callan, J. and Croft, W.B. 1994. Inquery system overview. *Proceedings of the TIPSTER Text Program (Phase I),* pp. 47-67. San Francisco, CA: Morgan Kaufman Publishers Inc.

Darwish, K. and Oard, D. 2003. Probabilistic structured query methods. *Proceedings ACM SIGIR*, Toronto, Canada, pp. 338-344.

Hersh, W., Price, S., Kraemer, D., Chan, B., Sacherek, L., and Olson, D. 1998. A large-scale comparison of Boolean vs. natural language searching for the TREC-7 Interactive Track.
The Seventh Text REtrieval Conference (TREC-7), Gaithesburg, MD. Available at: http://trec.nist.gov/pubs/trec7/t7_proceedings.html

Pirkola, A. and Järvelin, K. 2001. Employing the resolution power of search keys. *Journal of the American Society for Information Science and Technology*, 52(7): 575-583.

Appendix - the automatically selected expansion keys

The automatically selected expansion keys presented below are sorted by the score of word's relative frequency, i.e., word's frequency in a sample of documents relevant for Genomics Track's training topics divided by word's frequency in the same number of randomly selected documents. *Min* after many values show that in that case the real value could be higher (but not lower) than the value shown and used in the experiments as a basis of the selection of expansion keys. This is due to removal of words of low frequency (Section 2.3).

expression	9,3	level	3,9
sequence	9,3 (min)	mutant	3,9 (min)
regulate	9,1 (min)	membrane	3,8 (min)
signal	8,8 (min)	domain	3,7 (min)
induce	7,0 (min)	expressing	3,6 (min)
mutation	5,9	site	3,6 (min)
gene	5,7	receptor	3,6
transcription	5,4 (min)	function	3,5
activation	5,3	genetic	3,3
apoptosis	5,1 (min)	metabolism	3,3
cell	4,6	tyrosine	3,3 (min)
amino	4,5 (min)	beta	3,2
kinase	4,5 (min)	cytology	3,2 (min)
activate	4,4 (min)	region	3,2 (min)
binding	4,2	protein	3,2
pathway	4,1 (min)	alpha	3,1