Sheffield University and the *TREC 2004 Genomics Track*: Query Expansion Using Synonymous Terms

Yikun Guo, Henk Harkema, Rob Gaizauskas
University of Sheffield, UK
{guo, harkema, gaizauskas}@dcs.shef.ac.uk

1. Introduction

In this paper we describe our approach to the Ad Hoc Retrieval task of the TREC 2004 Genomics Track. This is a conventional searching task based on a 10-year subset of MEDLINE (about 4.5 million documents and 9 gigabytes in size) and 50 topics derived from information needs obtained via interviews of real biomedical researchers. We will also discuss the results of our submitted runs.

The hypothesis we want to test is whether the performance on this particular retrieval task can be improved by expanding queries with synonyms of the original query terms. We use the UMLS Metathesaurus, a comprehensive collection of controlled vocabularies in the biomedical domain, to identify query terms in topics and to determine their synonyms. Our approach is simple in the sense that we only consider synonyms of query terms and do not exploit hierarchical relations between terms such as hyponomy and hyperonymy.

Synonymy-based query expansion generally increases recall, but decreases precision due to ambiguous terms. Word senses of ambiguous terms which are inappropriate with regard to the topic under consideration give rise to “polluting” synonyms. We hope that the use of a specifically biomedical term resource such as UMLS will limit the negative effects synonymy-based query expansion may have on precision.

2. Methods

In this section we describe how we derive a set of query terms from a given topic and how this set is used by the Information Retrieval engine to find a set of documents.

2.1. Generating Query Terms

General Approach

Our general approach is to identify terms in a topic, where is term is understood to be a (multi-word) expression that is relevant in the domain under consideration. Given a set of terms for a topic, we expand this set by finding synonyms for these terms. The terms together with their synonyms will serve as the query terms that are submitted to the Information Retrieval engine.

To identify terms in topics and to find their synonyms we rely on the UMLS Metathesaurus (Humphreys et al., 1998). The UMLS Metathesaurus provides a semantic classification of terms from a wide range of vocabularies in the clinical and biomedical domain. It currently contains well over 2 million distinct English terms. Since the Ad Hoc Retrieval task involves searching over MEDLINE abstracts using topics derived from information needs of biomedical researchers, we assume that the UMLS Metathesaurus is a useful source of domain-relevant terms (see McCray et al. (2001) for more discussion of this issue).

To map a given topic to the Metathesaurus, we use the MetaMap program (Aronson, 2001). Using linguistic knowledge of various kinds, MetaMap identifies UMLS terms and variant forms of these terms in free text. For each term it finds, MetaMap will return one or more unique concept identifiers (CUIs) for that term.[1] Given the CUIs of a term, we look in the UMLS Metathesaurus for all other terms that are assigned at least one of these CUIs: these terms form the set of synonyms for the original term.

A quick inspection of the performance of MetaMap on the five sample topics provided with the data for the Ad Hoc retrieval task revealed that it did not identify all relevant terms in these topics. In particular, it did not pick up protein and gene names such as *p63* and *MTP1*. It also missed other potentially relevant terms, e.g., *tumorogenesis* and *inflammatory perturbation*. In order to remedy this problem, we employed a chunker (LT CHUNK[2]) to find NP chunks and symbols to be used as additional query terms. Symbols in LT CHUNK include single, non-numerical characters and certain punctuation marks that cannot be classified otherwise, as well as sequences of characters containing at least one alphabetic character and at least one numeric character. We only retained symbols of length greater than 1; these are assumed to be protein and gene names.

Using the procedure outlined above, we find, on average, 9.4 UMLS Metathesaurus terms per topic, and 9.2 LT chunks per topic.[3] Each UMLS term generates approximately 5.4 synonymous terms from UMLS.

Filtering

The UMLS Metathesaurus is a comprehensive resource, containing all sorts of terms. Not all of these terms are useful for text processing in general or for a given application in particular. For this reason we apply two sets of filters to potential search terms.

First, terms identified by MetaMap are ignored if they occur in a short, manually assembled list of stop words. This list contains words matched by MetaMap which are too general to be useful, e.g., *information*, *researcher*, *retrieve*, *literature*. We have also identified a set of semantic types (TUIs) in the UMLS Semantic Network which we *a priori* judged to be irrelevant to the domain. This list contains semantic types such as “Daily or Recreational Activity”, “Professional or Occupational Group”, and “Manufactured Object”. Any term such that all of its CUIs are assigned to irrelevant semantic types is ignored for further processing. Terms making it through the stop word filter and the irrelevant semantic type filter will be considered for synonym expansion.

The second set of filters applies to the set of synonyms generated for a given term. This set of filters is based on Aronson (2003). A synonym is removed in the following cases:

1. The synonym is a “bad” string, i.e., because of its form it is very unlikely to appear in a MEDLINE abstract. Strings that contain punctuation marks such as “=” and “!” are considered bad strings. Additionally, strings that contain classificatory elements such as “not elsewhere classified”, “not otherwise specified” are removed.

[1] We ran MetaMap using the default values for the Filtering Options and under the strict processing model (see http://mmtx.nlm.nih.gov/mmtx.shtml#Processing).

[2] See http://www.ltg.ed.ac.uk/software/chunk/index.html.

[3] We currently do not have any numbers regarding the overlap between UMLS terms and chunks.

2. The synonym belongs to a “bad” UMLS term type. Bad term types indicate strings that are inappropriate for text processing (see Aronson (2003) for further details).
3. The synonym is marked as a “suppressible synonym” in UMLS. Suppressible synonyms are usually short forms of terms that give rise to problematic cases of ambiguity, e.g., *Abdomen* is a suppressible synonym of *Malignant neoplasm of abdomen*.
4. The synonym consists of more than five words, e.g., *solute carrier family 11 (proton-coupled divalent metal ion transporters*).[4]
5. The synonym is string-identical to the term it is a synonym of.

Additional Information from UMLS

Besides supplying CUIs for identifying synonyms of terms, the UMLS Metathesaurus provides other information that can be used when preparing sets of query terms. The previous section already illustrated the use of semantic types from the UMLS Semantic Network to discard irrelevant terms. Semantic types can also be used in a positive way to mark up terms that are especially important to the domain under consideration. The Information Retrieval engine can use this mark-up to assign extra weight to the query terms concerned or to filter the document collection. We did not exploit this possibility in our current system.

One of the sources included in UMLS is the Medical Subject Headings controlled vocabulary of biomedical terms (MeSH). This enables us to recognize MeSH terms in topics and identify which of a term’s synonyms are MeSH terms.[5] This information can be used by the Information Retrieval engine for matching against the MeSH fields of MEDLINE citations.

2.2. Document Retrieval

Corpus Indexing

The document collection for the Ad Hoc retrieval task is a 10-year subset of the MEDLINE bibliographic database, amounting to roughly 9 gigabytes of textual data. In order to search efficiently in this document collection we use the Lucene text search engine.[6] Lucene consists of a set of APIs providing high-performance, full-featured ranked searching functionality, implemented in Java. Because of its high efficiency and cross-platform usability, it has been widely used in many applications to provide full text search functionality.[7]

One advantage of using Lucene is that one can specify different ways of indexing the various fields associated with a document. For example, adjunctive fields such as “PUBLISHING DATE”, “AUTHORS”, and “JOURNAL” etc., which are not relevant to the task, will be stored but not indexed and can be retrieved after a search. The contents of fields such as “TITLE”, “ABSTRACT” and “MESH”, however, will first go through a set of filters for stemming, removal of stop words, and tokenization, and will then be indexed. The whole index procedure requires just a few megabytes of memory. It runs with reasonable speed, about 350MB text per hour on an average Unix machine, for indexing the whole collection.

[4] This string is also considered a “bad” string because it contains a parenthesized expression.

[5] Synonyms that are MeSH terms are subject only to filter 5 mentioned in section 2.1.2.

[6] See http://jakarta.apache.org/lucene/docs/index.html.

[7] Cf. http://wiki.apcha.org/jakarta-lucene/PoweredBy.

Query Construction

After the query terms have been identified and expanded as described in section 2.1, these terms are used to construct a query to be matched against the documents in the collection. Lucene supports field search, where one can specify which fields a query will be matched against. For example, the query “title:mouse” will restrict Lucene to search for the keyword *mouse* in title fields only, ignoring other fields. Terms consisting of multiple tokens, e.g., *histoplasma capsulatum*, are searched using the phrase operator, ensuring that the string is matched exactly. Query terms can be combined with Boolean operators, such as “AND”, “OR” and “NOT” to form complex queries. In our experiment, multiple terms from the same topic are combined into one query using the “OR” operator. One of the factors determining the relative ranking of a document is the number of disjuncts in a complex query matching the document, where more matching disjuncts implies a higher rank. In addition, each query term can also be given a boosting factor to reflect its relative importance. Lucene will bias its search to those queries with high boosting factors. The relevant documents are returned in a descending order.

In query construction, all three parts of a topic, i.e., title, information need and context, are treated equally and used to provide search terms. After merging repeated instances of terms, the terms are searched against both the title and abstract field of each MEDLINE abstract, except for the MeSH terms – these are searched against the MeSH field. Since we found that some noun phrases such as “protein”, “genes” etc., are too general in the sense that searching for these terms retrieves too many documents, queries that return more than 10K documents are filtered out in the last step of the query construction process. This will prevent relevant matching documents from being overwhelmed by non-relevant matching documents.

The queries used in the first run consist of terms identified by MetaMap and LT CHUNKER. For the second run, synonyms are added to the queries, using the “OR” operator. The queries containing the original keywords are given a boosting factor of 5, while the queries for the synonyms are given a boosting factor of 1. These factors were determined from experimentation with the five sample topics.

Searching the document collection with Lucene is very fast. Given a query, Lucene will first pass it through the same filters that are used in the indexing step to do stemming, stop word removal, and tokenization. Next, it will retrieve a set of documents and compute their relevance scores. Typically, the whole process takes no more than 20 seconds per query.

3. Results and Discussion

We submitted two runs for the Ad Hoc Retrieval task. The first run is our baseline, where no synonyms were added to the query. The second run is with synonyms. Our hypothesis is that performance will improve by expanding queries using synonyms from UMLS. However, the various evaluation statistics indicate that performance does not differ very much between the two runs. As tables 1 and 2 show, the results of the second run are just slightly better than those of the first run. The non-interpolated average precision over all relevant documents for run 1 is 0.1294 vs. 0.1304 for run 2. Run 1 achieved an exact R-precision of 0.1632; the R-precision of run 2 is 0.1619. The total number of relevant documents retrieved over all topics is 2228 for run 1 and 2402 for run 2 (out of a total number of 8268 relevant documents).

We also compared our results against the per-topic best, worst, and median scores of the participants; the performances of the two runs are not very good in terms of the average precision:

only 10 and 9 topics out of the 50 topics making up the task are above the median respectively. However, with regard to the precision at 10 documents, our system does relatively well: there are 23 topics above the median and 5 of these are the best (they all achieved a precision of 1.0) for the second run. This suggests that our system performs relatively well at high precision side in the recall-precision graph. Graph 1 shows the interpolated recall-precision of our runs. Note that high precision at 10 documents is an important factor for information retrieval systems, where users will typically only inspect the top 10 documents.

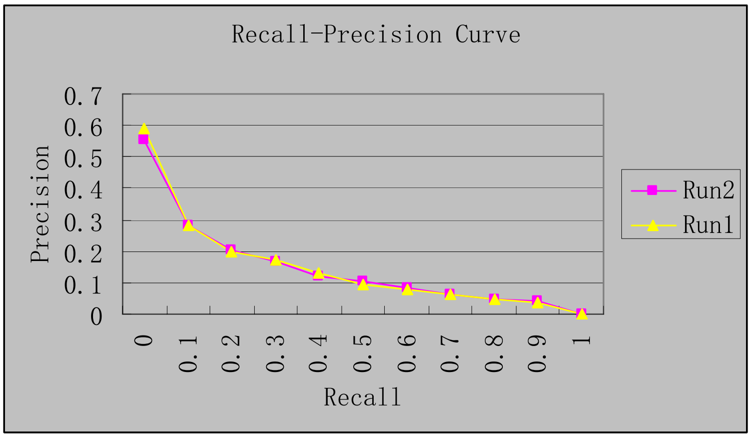
Recall	Precision run 1	Precision run 2
0.00	0.5898	0.5545
0.10	0.2799	0.2806
0.20	0.1959	0.2032
0.30	0.1717	0.1676
0.40	0.1297	0.1209
0.50	0.0932	0.1064
0.60	0.0788	0.0811
0.70	0.0605	0.0630
0.80	0.0454	0.0496
0.90	0.0365	0.0392
1.00	0.0021	0.0000

Table 1: Recall level precision averages for run 1 and run 2

At *x* docs	Precision run 1	Precision run 2
5	0.4160	0.3840
10	0.3540	0.3660
15	0.3400	0.3480
20	0.3250	0.3270
30	0.2940	0.2913
100	0.1892	0.1850
200	0.1300	0.1329
500	0.0712	0.0735
1000	0.0446	0.0480

Table 2: Document level averages for run 1 and run 2

To analyze our results further, we grouped the query terms into three classes: 1) chunk terms obtained from the output of LT CHUNKER, 2) UMLS terms recognized by MetaMap and, 3) synonyms of the UMLS terms. We used each of these three classes of keywords to search directly against the sets of true relevant abstracts in order to count how many abstracts are retrieved. The resulting numbers can be viewed as the upper bound for our approach and can be used to evaluate the relative importance of each class of query terms. The results of this exercise are summarized in table 3. For each class of terms this table shows the total number of true abstracts retrieved using this class of terms only (# abstr. matched), the number of terms of this class identified in a topic (# terms), and the number of abstracts matched per term (ratio, i.e., (# abstr. matched) / (# terms)). All numbers are averaged over the 50 topics. For example, on average, the set of UMLS terms found in a topic matches 145.0 true relevant abstracts.



Graph 1: Interpolated Recall-Precision Averages of our two runs

Furthermore, we established that, averaged over all 50 topics, the set of UMLS terms and chunk terms for a topic matches 152.4 true relevant documents, whereas the set of synonyms for a topic only matches 5.0 additional documents, i.e., documents which are not already matched by the UMLS terms and the chunk terms.

	# abstr. matched	# terms	ratio
Chunk terms	111.8	9.4	11.9
UMLS terms	145.0	9.2	15.8
Synonyms	131.9	49.8	2.6

Table 3: Matching against true abstracts

We also studied the contribution of each of the three classes of terms in the context of the full retrieval task rather than their matching power against the set of true relevant documents. The results are shown in table 4: for each class of terms it provides the interpolated recall-precision averages over all queries, where these queries consist of terms from the class specified by the column only.[8] It is interesting to note that while the UMLS terms retrieve more relevant documents (2270) than the chunk terms (1838), using the latter class of terms produces higher precision scores at the first few recall levels. Using synonyms only returns 1057 relevant documents and precision levels are poor in this case. Comparing tables 1 and 4 we see that none of the three classes of terms used on their own outperforms run 1 (chunk terms + UMLS terms) or run 2 (chunk terms + UMLS terms + synonyms).

[8] Note that the set of synonyms of a given term does not include the term itself.

Recall	Prec. chunk terms	Prec. UMLS terms	Prec. synonyms
0.00	0.5545	0.4673	0.1076
0.10	0.2435	0.1993	0.0301
0.20	0.1700	0.1445	0.0233
0.30	0.1171	0.1133	0.0078
0.40	0.0685	0.0933	0.0056
0.50	0.0489	0.0752	0.0043
0.60	0.0355	0.0550	0.0040
0.70	0.0194	0.0399	0.0036
0.80	0.0107	0.0173	0.0001
0.90	0.0017	0.0033	0.0001
1.00	0.0002	0.0000	0.0001

Table 4: Recall-precision averages for chunk terms, UMLS terms, and synonyms

This preliminary analysis reveals that both UMLS terms and chunk terms play an important role in retrieving relevant abstracts. Synonymous terms seem to be less important: many synonyms are found, but only a few new relevant abstracts are retrieved by these terms. Maybe the synonyms in UMLS are not very relevant to this particular task. A better query expansion method might be to add the highly weighted terms found in the top ranked retrieved abstracts to a query and search again. Some further experiments are needed to verify this.

4. Conclusion

In this paper we have described our approach to the Ad Hoc Retrieval task of the TREC 2004 Genomics Track. We submitted two runs: one baseline run and one run in which the query terms from the baseline run were expanded with synonymous terms. To find query terms in topics and to determine synonymous terms, we made use of the UMLS Metathesaurus.

It turns out that recall computed over the fixed return sets of 1000 documents for each query goes up when synonyms are included, but only slightly: 2186 relevant documents retrieved in the first run vs. 2346 relevant documents retrieved in the second run (out of a total of 8268 relevant documents). Precision computed over the fixed return sets increases very slightly, too.

The various evaluation measures indicate that the rankings produced by the two runs are virtually the same: the use of synonyms promotes neither relevant documents nor irrelevant ones in the ranking. Further analysis of the results shows that even though each UMLS search term generates about 5.4 synonyms, these synonyms are less “powerful” than the original search terms. On average, each synonym matches only 2.6 relevant documents, whereas the original search terms match 11.9 documents (for UMLS terms) and 15.8 documents (for chunks). Also, the expanded queries retrieve only very few fresh documents, i.e., relevant documents that were not retrieved by the original, unexpanded queries. This suggests that the document collection and the topic set in this particular experiment exhibit a shared vocabulary: generally, a document relevant to a given topic contains the same terms as the topic rather than terms synonymous to terms in the topic.[9]

[9] Our experimental set-up does not allow us to make any claims about the incidence of relevant

Apparently, the increase in the relevance scores of relevant documents that do contain synonyms is neutralized by a similar increase in the relevance scores of irrelevant documents containing synonyms, leaving the overall ranking of the retrieved documents unchanged. Alternatively, the boosting factor assigned to synonyms, which was derived from a very small and incomplete training set, might be too low for these terms to have a noticeable effect.

The poor performance of queries consisting exclusively of synonyms indicates that the use of synonyms draws many irrelevant documents into the result set. In this light, it is rather surprising that run 2 (with synonyms) does not perform worse than run 1 (without synonyms). Again, this may have to do with the boosting factor assigned to synonyms. The poor performance of synonyms can also be taken to show that the use of synonymous terms within documents is limited. In the theoretical case in which a document contains all synonyms of the terms occurring in it, replacing the terms in a topic with their synonyms would not affect retrieval performance negatively. Of course, this theoretical situation does not obtain. However, since the documents used in the task are MEDLINE abstracts, which tend to be relatively short and non-repetitive, thus limiting the opportunity for authors to use synonyms, this argument is worth considering.

Future work will focus on increasing the baseline performance, which will allow us to draw firmer conclusions regarding the observed minimal effect of the use of synonyms. We will also investigate to what extent the use of synonyms contributes to falsely retrieved documents. Furthermore, we will connect our results to earlier, similar work.

Acknowledgements We would like to thank Angus Roberts for his help with setting up and running MetaMap and LT CHUNKER for us. We would also like to acknowledge the support of the UK Engineering and Physical Sciences Research Council (grant number: GR/R67743) and Medical Research Council (grant number: 60086) which has made possible the work reported here.

References

A.R. Aronson. 2001. Effective Mapping of Biomedical Text to the UMLS Metathesaurus: the MetaMap Program. In: *Proceedings of the American Medical Informatics Association Symposium*, pp. 17-21.

A.R. Aronson. 2003. *Filtering the UMLS Metathesaurus for MetaMap.* http://skr.nlm.nih.gov/papers/index.shtml.

L. Humphreys, D.A.B. Lindberg, H.M. Schoolman, and G.O. Barnett. 1998. The Unified Medical Language System: An Informatics Collaboration. In: *Journal of the American Medical Informatics Association*, 1(5): 1-13.

A.T. McCray, O. Bodenreider, J.D. Malley, and A.C. Browne. 2001. Evaluating UMLS strings for Natural Language Processing. 2001. In: *Proceedings of the American Medical Informatics Association Symposium*, pp. 448-452.

documents that contain terms whose synonyms (that do not occur in the document as terms themselves) match synonyms of terms occurring in topics. (Note that the existence of a situation in which a synonym of a term in a document matches a term in a topic implies the existence of a situation in which a synonym of a term in a topic matches a term in a document.)