Knowledge-intensive and statistical approaches to the retrieval and annotation of genomics MEDLINE citations

Alan R. Aronson,[a] Dina Demner,[a,b] Susanne M. Humphrey,[a] Nicholas C. Ide,[a] Won Kim,[a] Hongfang Liu,[c] Russell R. Loane,[a] James G. Mork,[a] Lawrence H. Smith,[a] Lorraine K. Tanabe,[a] W. John Wilbur,[a] Natalie Xie[a]

[a]National Library of Medicine, Bethesda, Maryland
{alan, demner, humphrey, ide, loane, mork}@lhc.nlm.nih.gov;
{wonkim, lsmith, tanabe, wilbur, natxie}@ncbi.nlm.nih.gov

[b]University of Maryland, College Park, Maryland
[c]University of Maryland, Baltimore County, Maryland
hfliu@umbc.edu

Abstract

Retrieving and annotating relevant information sources in the genomics literature are difficult but common tasks undertaken by biologists. The research presented here addresses these issues by exploring methods for retrieving MEDLINE® citations that answer real biologists' information needs and by addressing the initial tasks required to annotate MEDLINE citations having genomic content with terms from the Gene Ontology (GO). We approached the retrieval task using two methods: aggressive, knowledge-intensive query expansion and text neighboring. Our approaches to the triage subtask for annotation consisted of traditional machine learning (ML) methods as well as a novel ML algorithm for thematic analysis. Finally, we used a statistical, n-gram heuristic to decide which of the GO hierarchies should be used to annotate a given MEDLINE citation.

Keywords: Genomics; MEDLINE; MeSH; Information Retrieval; Vector Space Models; Statistical Natural Language Processing; Machine Learning; Thematic Analysis; Decision List Learning.

1 Introduction

The need for improved access to genomic information stored in bibliographic and structured databases is well-established. One venue for focusing research to answer this need is the Genomics Track of the Text Retrieval Conference. As in 2003, the National Library of Medicine® (NLM®) and University of Maryland (UMd) teamed up to participate in the track at TREC-13 in 2004.

This year the Genomics Track had an ad hoc retrieval task and a categorization task. We explored two approaches for the ad hoc retrieval task. One approach was based on the same search engine (SE) we used last year and applying several knowledge intensive methods to formulate SE queries by extracting information from the task queries. The second approach used TexTool, a text neighboring variant of the PubMed Related Articles algorithm which is based on tf*idf vector retrieval. Both are discussed in Section 2. We applied several ML approaches to the triage subtask of the categorization task as well as a novel method for theme detection (Wilbur 2002); and we used two variants of a decision list learning algorithm to address the annotation hierarchy subtask. The categorization task is described in Section 3. Finally, we provide some conclusions of our research in Section 4.

2 Ad hoc Retrieval Task

The novelty of the ad hoc task this year was in the attempt to address real users' needs. These needs were solicited in interviews with biologists, and documented in traditional TREC topic format. This topic presentation is considerably different from the first year of the track that provided only highly relevant search terms, i.e. gene and organism names.

2.1 SE query formulation

Building on the last year's experience (Kayaalp et al., 2003), we identified gene names and organisms as important entities to be extracted from the topic statements. Recognition of genetically oriented citations in the document collection was the second major factor in improvement of our last year's results. In addition, we used knowledge intensive and statistical methods to find phrases and terms specific to the topic, and represented in MeSH (Aronson 2001). The extracted information was used to prepare the final

query as an XML document that was then translated into SE syntax and weighted. (SE is a retrieval search engine developed at NLM specifically for biomedical text. See Kayaalp et al., 2003.) Collection indexing, and scoring of the retrieved documents did not differ from those used last year.

2.1.1 Gene name recognition

We used ABGene, a gene and protein name tagger trained on MEDLINE abstracts (Tanabe and Wilbur 2002) to identify gene names in the TITLE, NEED and CONTEXT fields of the topics. We did not preprocess the topic, or change the program itself, with the exception of capitalizing the first word in each sentence, so that the input to ABGene looked more like the input expected by this tool. ABGene applies sophisticated rules to distinguish between potential gene names and gene name-like strings, which is necessary when processing MEDLINE abstracts. We relaxed these requirements externally under the assumption that if the topic provides a gene-name like term, e.g. mixed case words, it either is a gene name, or an important term that should be included in the query. Using this rule we extracted many key terms in addition to names recognized by ABGene, e.g. in topic 14, "Expression or Regulation of TGFB in HNSCC cancers" TGFB was recognized by ABGene, and HNSCC was recommended by the rule.

2.1.2 Organism Identification

The task of populating the organism field in our structured query was twofold: first organisms had to be found in the topic, if present, and then had to be mapped to exact indexing terms, i.e. the corresponding MeSH descriptor and/or Check Tag. We used the NCBI taxonomy database located at the site http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=Taxonomy to find an organism in the topic text as follows: organism's Latin name, synonyms and common names were extracted from the taxonomy, Latin names were mapped to MeSH terms using MeSH tree files. Topics were searched for each synonym of each of the model organisms until the first match. In case of the match the corresponding MeSH term was added to the query.

2.1.3 MeSH term identification

MeSH terms were identified in the TITLE, NEED and CONTEXT fields of the topic using MetaMap, and subsequently ranked using tf*idf. The high ranking terms were added to the query. Because of missing synonymy some of the high ranking phrases were not recognized as MeSH terms. Using the TexTool Theme Query we were able to get the missing synonyms, e.g. phrase "subcellular fractionation" was mapped to 'Subcellular Fractions'.

2.1.4 Journal descriptor indexing

A broad indexing method known as Journal Descriptor Indexing (JDI) (Humphrey et al., 2000) was used as a broad filter to eliminate about 61% of the test collection from further consideration by identifying the remaining 39% as likely to be of genetic interest. This method of identifying documents in the genetics domain was explored in previous research (Hristovski et al., in press) for the purpose of gene symbol disambiguation. For example, determining that a document title "Ethics in a twist: 'Life Support' BBC1" is outside the genetics domain serves to disambiguate BBC1 -- the British television station, as in this title, from the breast basic conserved 1 gene.

2.2 TexTool

For our second retrieval approach, we concatenated the TITLE field with itself and the NEED and CONTEXT fields to create a surrogate document for each query. A program called TexTool was applied to the surrogate documents to find 1,000 similar documents for each query. TexTool is a text neighboring variant of the PubMed Related Articles algorithm, a vector retrieval method well suited to comparing documents that have bag-of-words form (an unordered set of words each coupled with its frequency). If the term t occurs in the document d, it has been found very useful to define v_{dt} as the product of two weights

$$v_{dt} = lw_{dt} \cdot \sqrt{gw_t} \qquad (1)$$

If tf_d denotes the frequency of the term t within document d and $dlen$ denotes the length of d (sum of all $t'f_d$ for all t' in d) then we define the local weight in

$$lw_{td} = 1/\left(1+\exp\left(\alpha \cdot dlen\right)\cdot \lambda^{tf_d - 1}\right) \qquad (2)$$

where $\alpha = 0.0044$ and $\lambda = 0.7$.

This formula is derived from the Poisson model of term frequencies within documents and has been found to give good performance on MEDLINE documents. It is common to define the global weight, gw_t as the inverse document frequency (idf)

$$gw_t = \log_2\left(N / f_t\right) \qquad (3)$$

where N is the total number of MEDLINE documents in the database and f_t is the number of documents in the MEDLINE database which contain the term t. Given documents q and d the document

similarity is given by the inner product of two vectors:

$$sim(q,d) = v_q \cdot v_d \qquad (4)$$

The TexTool method generally worked best for queries that contain the gene of interest in all three fields (TITLE, NEED and CONTEXT), and for queries that contain relevant phrases in the CONTEXT field. For example, it performed well over the median for query #22, which contains the term *p53* in all three fields, and includes the following important phrases in the CONTEXT field only: *DNA damage, cell cycle arrest,* and *apoptosis*. Interestingly, the method also tended to perform above the median on general questions without specific gene names. For example, it performed far above the median on query #33:

<TITLE>Mice, mutant strains, and Histoplasmosis</TITLE>
<NEED>Identify research on mutant mouse strains and factors which increase susceptibility to infection by Histoplasma capsulatum.</NEED>
<CONTEXT>The ultimate goal of this initial research study, is to identify mouse genes that will influence the outcome of blood borne pathogen infections.</CONTEXT>

For broad topics like these, using all of the information available in the three fields may be more important than attempting to focus the query.

One of the method's shortcomings is that it always returns 1000 documents. This is problematic for queries that are associated with very few relevant documents. The method performed far below the median on query #18, which has only one relevant document associated with it. The method also performed poorly on gene-specific queries that do not include the gene name in the CONTEXT field. For example, the method performed far below the median for query #36. In this query, the gene *RAB3A* is included in the TITLE and NEED fields, but not in the CONTEXT field. Query #36 illustrates an additional weakness of this method, and of vector retrieval methods in general - non-gene-related phrases can minimize the contribution of gene names for retrieval. For example, in Query #36, the CONTEXT field contains the terms *synaptic plasticity*, *learning* and *memory*. These terms occur in many documents without any reference to genes and/or proteins, and the related documents retrieved reflect this tendency.

2.3 Ad hoc task results

SE performance this year was uneven with 20 out of 50 topics at or above median, mean average precision of 0.2191, and R-precision of 0.2351. Interestingly, exclusion of the probably relevant (PR) documents from relevance judgments[1] did not change the numbers much (difference in both measures in the third decimal digit), i.e. most of the documents retrieved by SE are definitely relevant. Since the whole title field of the topic was used with the highest weight as part of the weighted query, it is not surprising that retrieval results are better for the topics where the title field was modeled as a short query that an experienced user might use in a web search, e.g. topic 35 (WD40 repeat-containing proteins). On the contrary, when the query was too broad, e.g. topic 11 (Carcinogenesis and hairless mice), even shorter queries that had better results with all other factors being equal, did not perform well. Specific genes identified in other topic fields could not compensate for the effect of the underspecified title, e.g. terms "apolipoprotein e4" and "factor v mutations" identified by ABGene did not improve performance on topic 38 (Risk factors for stroke). There seems to be no correlation between presence of an identified organism in the specific field of the query and performance on the topic. This observation could be verified in the experiment, where organism fields in the xml query are omitted. In several cases only few documents with probability above threshold were retrieved. In five of these cases the number of documents identified as relevant exceeds the number of documents retrieved by SE. This might be the consequence of tuning parameters on a very small training set. Another assumption that was confirmed by the training set, and hurt the test performance was reliance upon presence of highly specific terms, such as gene names in the title field. As a matter of fact, when the assumption holds, SE achieves median and above performance, i.e. for all 20 topics which fall into this category.

The TexTool run (tq) achieved an average precision of 0.2277, and an R-precision of 0.2877. It performed at or above the median for 34/50 queries. Figure 1 compares tq results with median results as well as those for the SE run (see).

[1] http://trec.nist.gov/act_part/tracks/genomics/04.readme.txt defines the relevance judgments for the ad hoc task.

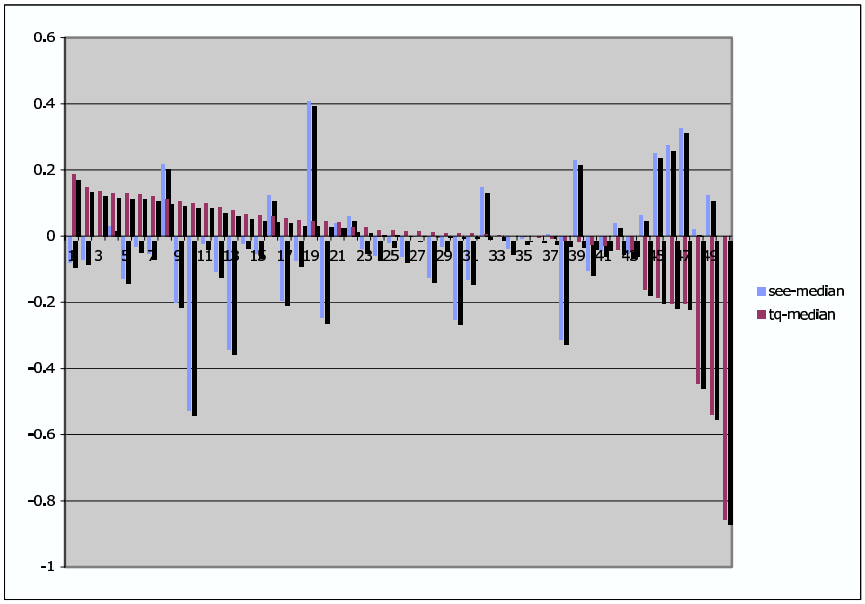

Figure 1. Comparison of TexTool (tq) and SE (see) runs with median results, ordered by decreasing tq-median values

3 Categorization Task

The NLM/UMd team explored the triage and hierarchical annotation subtasks of this year's categorization task aimed at annotating MEDLINE citations with appropriate GO terms. The following sections describe these subtasks.

3.1 Triage subtask

We took five approaches to the triage subtask: three standard machine learning algorithms, Bayes, Support Vector Machines (SVM) and AdaBoost, performing 3-fold cross validation on the training set for each algorithm, and two variations of a novel approach based on finding themes within text (Wilbur 2002).

3.1.1 ML methods

Naïve Bayes: The naïve Bayesian algorithm is based on the assumption that the values of attributes are distributed independently within the classes to be learned. Thus each term can be weighted separately based on its distribution in the training set. One scores documents in the set by summing the weights of the terms they contain and then ranks the documents based on the resultant scores. For details the reader may consult (Langley, Iba et al. 1992; Langley and Sage 1994; Langley 1996).

Support Vector Machine (SVM): Define the loss function

$$h(z)=\begin{cases}|1-z|, & z<1\\ 0, & 1\le z\end{cases} \qquad (5)$$

Assume we are given training data $\{(\vec{x}_i, y_i)\}_i$ where y_i is 1 or –1 depending on whether the data point $\vec{x}_i$ is classified in the + or the – class. For SVM one seeks that vector $\vec{w}$ that minimizes

$$\sum_i h(y_i \vec{x}_i \cdot \vec{w}) + \lambda \|\vec{w}\|^2 . \qquad (6)$$

We solve this problem using Platt's sequential minimal optimization algorithm (Platt 1998; Platt 1999). We have found $\lambda = 5$ to work well ($C = 0.1$ in the usual treatment (Burges 1999; Platt 1999)).

Boosted decision trees: We use an improved version of AdaBoost (Schapire and Singer 1999) and boost binary decision trees where the splitting criterion is designed to minimize the error limit computed in AdaBoost. We have examined trees of depths 1-5 and have found depth 4 to give the best results, and that is what we report here. There is also no set limit to the number of rounds of boosting to use in learning. However, in examining many rounds of boosting, one generally sees improvement early and then a more or less stable performance with some oscillation which appears due to noise. We determined the number of rounds of boosting which gave the best average performance in the first 1000 iterations in the training set.

3.1.2 Thematic analysis method

Thematic analysis uses an Expectation-Maximization (EM) algorithm to construct a theme which consists of a set of words, U (of fixed cardinality), and a subset, V, of the set of training documents, D. The algorithm, shown pictorially in Figure 2, is seeded with a document subset (the set of positive training documents). From V, the set of words U that makes V, D-V most probable is selected. The words in U are given Bayesian weights, and all of the documents in D are scored. A new V is found by taking documents in D scoring above a threshold. The algorithm proceeds until U and V converge to a theme which can be used to score test documents. For the triage task, we kept track of where the words in U occurred (title, gloss, introduction, materials and methods, etc.)

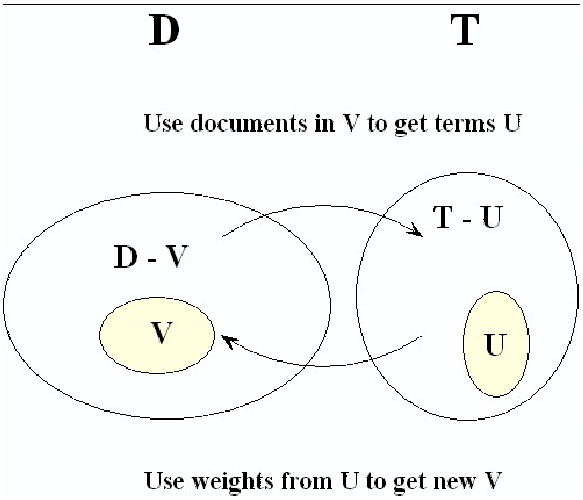

Figure 2. Thematic analysis algorithm

We applied the theme generation algorithm to the triage training data using two methods: (1) where the training set was used to generate one theme T, and (2) where the data were separated according to journal to generate three journal-specific themes. The motivation for (2) came from the observation that sub-terminologies existed for each of the three journals due to the different perspectives of molecular biology, biochemistry and general science. We reasoned that we could use these sub-terminologies to get a more specific picture of articles in each journal that were selected for GO annotation. We performed 5-fold (respectively, 3-fold) cross validation on the methods using a Bayesian score cutoff of zero.

3.1.3 Triage subtask results

The following tables present the results of the triage subtask

Run	Precision	Recall	F-score	Normalized Utility
NLMT2SVM	0.1286	0.7333	0.2188	0.4849
NLMT2BAYES	0.0902	0.8690	0.1635	0.4308
NLMT22	0.1986	0.4810	0.2811	0.3839
NLMT21	0.1950	0.4643	0.2746	0.3685
NLMT2ADA	0.0713	0.9881	0.1330	0.3448

Table A. Five official runs were submitted for the triage task: Machine learning methods (NLMT2SVM, NLMT2BAYES and NLMT2ADA), one theme (NLMT21) and three journal-specific themes (NLMT22).

All Training Data			JBC Only		
Wt	Term	Section	Wt	Term	Section
1.54	mice	Text	1.48	mouse	Results
1.38	mouse	Results	1.67	mouse	Text
1.51	mouse	Text	1.64	mice	Text
1.25	mice	Results	1.14	mice	Introduction
1.13	mice	Introduction	1.30	mice	Results
1.09	mice	Discussion	1.62	Mouse	Results
1.05	mouse	Introduction	1.14	mouse	Introduction
1.31	mice	Figure	2.11	Mouse	Title
1.48	embryonic	Results	2.00	Mouse	Text
1.89	littermates	Results	1.27	Mouse	Introduction
1.57	Mouse	Results	1.00	mice	Discussion
1.68	heterozygous	Results	1.21	liver	Results
1.37	mice (Fig	Results	1.73	embryos	Results
2.09	Mouse	Title	1.50	mice (Fig	Results
1.95	targeting	Materials/Methods	1.36	embryonic	Results
1.06	mice	Materials/Methods	1.53	spleen	Results
1.18	liver	Results	1.05	tissues	Results
1.20	Mouse	Introduction	1.95	littermates	Results
2.44	targeting vector	Materials/Methods	1.32	heart	Results
1.92	Mouse	Text	0.85	mouse	Discussion

Table B. The top 20 terms from themes generated from all positive and negative training data and JBC alone are similar. Wt = Bayesian weight, Term = theme term, Section = section of text where the theme term appears.

PNAS Only			JCB Only		
Wt	Term	Section	Wt	Term	Section
2.23	mice	Text	4.22	wild-type litter-mates	Results
2.86	embryonic stem	Materials/Methods	2.68	littermates	Results
2.81	targeting vector	Materials/Methods	2.56	apoptotic	Figure
2.43	recombination	Materials/Methods	1.94	mice	Discussion
2.43	targeting	Materials/Methods	1.95	paraformaldehyde	Materials/Methods
2.68	blastocysts	Materials/Methods	3.03	homozygous	Figure
2.19	exon	Materials/Methods	3.99	fertile	Results
1.91	mice	Materials/Methods	3.99	mouse embryos	Materials/Methods
2.21	stem	Materials/Methods	3.99	histological	Results
2.66	homologous re-combination	Materials/Methods	3.99	recombination	Figure
1.70	mice	Figure	2.80	genotype	Materials/Methods
1.67	mice	Results	2.62	survival	Figure
2.16	homologous	Materials/Methods	1.80	mice	Figure
1.59	Mice	Materials/Methods	1.75	mice	Materials/Methods

1.76	Southern	Materials/Methods	1.93	tissues	Results
1.77	embryonic	Materials/Methods	1.86	embryonic	Introduction
1.63	mouse	Materials/Methods	2.83	ES	Gloss
2.80	exon 2	Materials/Methods	2.83	ES	Materials/Methods
1.55	mice	Discussion	2.28	embryonic	Gloss
1.96	Southern blot	Materials/Methods	2.28	J	Results

Table C. The top 20 terms from themes generated from PNAS and JCB data only differ greatly from Top 20 terms derived from all training data. Wt = Bayesian weight, Term = theme term, Section = section of text where the theme term appears.

3.1.4 Triage subtask discussion

The results show that two of the machine learning methods, SVM and Naïve Bayes, outperformed the theme-based methods based on normalized utility scores. The theme-based methods did better than machine learning in terms of F-score. This suggests that a combined approach would improve performance overall. The low recall scores of the theme-based methods are related to the small number of training documents, which contain only a subset of the terminology used by annotators to determine GO relevance. The utilization of text section information contributes to the higher precision scores obtained by the theme-based methods. Three journal-specific themes consistently outperformed one combined theme in precision, recall, F-score and normalized utility. This may stem from the fact that the training data are skewed towards one journal, JBC (see Table B). In the one-theme method, important terms in the materials and methods and figure sections in PNAS and JCB are replaced by terms in the results sections of JBC articles (see Table C), causing a decrease in triage performance for PNAS and JCB articles. The results indicate that separating the training data by journal results in themes that more accurately depict the types of terms that are likely to signify GO relevance in each journal.

3.2 Annotation hierarchy subtask

The primary goal of this subtask is to correctly identify, given the article and gene name, which of the GO hierarchies have terms within them that have been annotated. The method used here is a heuristic based on statistical information of n-grams. It was motivated by the fact that local context around the given genes provides hints on determining the categories. The method contains three steps: preprocessing, training, and assignment. In the following sections, we provide detailed descriptions of each step.

3.2.1 Annotation hierarchy preprocessing

For each article, we extracted pure text from the SGML markup text. Note that some gene names provided in the (article, gene name) pairs did not appear in the text, so we applied a synonym list which was extracted from several online resources including iProClass, SwissProt, TREMBL, and MGI for the given MGI records. After changing to lower case and removing punctuation, the text was then tagged with the given MGI records. We extracted sentences that contained the corresponding MGI genes as well as one sentence before and one sentence after. Each occurrence of these three sentences was considered as a sample.

3.2.2 Annotation hierarchy training

We assumed that there was a list of n-grams which could signify the corresponding GO hierarchies. In the training stage, we tried to assign scores to pairs (ngram, hierarchy) so that significant n-grams would have a higher score. Let the total number of training examples be N. For n-gram g (where n is from 1 to 4) and GO hierarchy d, we assigned a score to the pair (g,d) according to the following formula:

$$Score(g,d) = \frac{F(g,d)}{F(g)} \times \left(\sum_t \left| \frac{F(t)}{N} - \frac{F(g,t)}{F(g)} \right|\right) \quad (7),$$

where $F(.)$ is the number of training samples that have the given values and t denotes GO hierarchies. For example, $F(g)$ denotes the number of training samples that contain n-gram g. We then selected n-grams with an occurrence that is larger than five.

3.2.3 Annotation hierarchy assignment

The assignment of the hierarchy to the pair (A,G), where A refers to articles and G refers to MGI genes, is based on samples extracted from the article for the

given gene. Given a sample S and a hierarchy d, we computed a score, $Score(S,d)$, which is the summation of scores of all n-grams g occurring in the sample, see the following formula.

$$Score(S,d) = \sum_{g} Score(g,d) \quad (8).$$

Then we computed a score for the tuple (A,G,d) according to the following formula

$$Score(A,G,d) = \max_{S} imum(Score(S,d)) \quad (9).$$

The assignment of the hierarchy d to the pair (A,G) was also based on the following assumption: if an article is GO annotated for one gene, then every gene mentioned in that paper will be GO annotated.

After obtaining $Score(A,G,d)$, we then sorted the articles according to $MAX(A)$, which is the maximum score for all genes G mentioned in the article:

$$MAX(A) = \max_{G,d} imum(Score(A,G,d)) \quad (9)$$

A list of articles was then selected according to the distribution in the training set. (We assumed that the test set and the training set have the same distribution regarding the number of articles that are GO annotated and the number of genes that are so annotated.) For each pair (A,G), a hierarchy d was assigned when the ratio of $Score(A,G,d)$ and $MAX(A)$ is larger than 0.8.

3.2.4. Annotation hierarchy results

We submitted two runs for the annotation hierarchy subtask. The first run was obtained using the provided training set. The second run employed a bootstrapping method, and was based on the result of the first run: for selected articles in the test set with a relatively high $MAX(A)$ in the first run, we considered the category assignment for them were reliable; we merged them with the training set to recompute n-gram scores and redo the hierarchy assignment.

Table D shows the results of the two submissions. As the table shows, we found that the inclusion of samples from the test set did not improve the performance. One reason is that the performance of the first run is not good enough for using bootstrapping methods. Note that our approach to the annotation hierarchy subtask was based on the assumption that if one of the genes in an article was GO annotated, then all genes in that article were GO annotated. In reality, this assumption is not true. The proposed method can be used without the assumption, in which case we need to establish thresholds for assigning GO hierarchies.

	Submission 1 (no bootstrapping)	Submission 2 (with bootstrapping)
Precision	0.4306	0.4270
Recall	0.5515	0.5374
F-Measure	0.4836	0.4758
Normalized Utility	0.5151	0.5013

Table D. The results for the two annotation hierarchy subtask submissions.

4 Conclusions

We recognize the need for realistic genomics tasks but felt that both tasks could have benefited from more training data, and the categorization task would have been easier to perform with additional information such as more detailed descriptions of curation policies. It is likely that combining methods for the various tasks would improve results. Even though both of the ad hoc query methods (SE and TexTool) did poorly on more general queries, the complementary performance of the methods exhibited in Figure 1 makes exploring a combined method a reasonable approach for the future. A similar observation made earlier for the categorization triage subtask also supports a future effort to combine methods.

References

Aronson, A. R. (2001) "Effective mapping of biomedical text to the UMLS Metathesaurus: the MetaMap program." *Proc AMIA Symp.*, 17-21.

Burges, C. J. C. (1999). *A tutorial on support vector machines for pattern recognition*, Bell Laboratories, Lucent Technologies.

Hristovski, D., Peterlin, B., Mitchell, J. A. and Humphrey, S. M. "Using literature-based discovery to identify disease candidate-genes." *International Journal of Medical Informatics*. In press.

Humphrey, S. M., Rindflesch, T. C., & Aronson, A. R. (2000). "Automatic Indexing by Discipline and High-Level Categories: Methodology and Potential Applications." Proceedings of the *11th ASIST SIG/CR Classification Research Workshop*, 103–116.

Kayaalp, M., Aronson, A. R., Humphrey, S. M., Ide, N. C., Tanabe, L. K., Smith, L. H., Demner, D., Loane, R. R., Mork, J. G. and Bodenreider, O. (2003)

"Methods for accurate retrieval of MEDLINE citations in functional genomics." *The Twelfth Text Retrieval Conference, TREC-2003*, Gaithersburg, MD: 441-50.

Langley, P. (1996). *Elements of Machine Learning*. San Francisco, Morgan Kaufmann Publishers, Inc.

Langley, P., W. Iba, et al. (1992). "An analysis of Bayesian classifiers." *Tenth National Conference on Artificial Intelligence*, San Jose, AAAI Press.

Langley, P. and S. Sage (1994). "Induction of selective Bayesian classifiers." *Tenth Conference on Uncertainty in artificial intelligence*, Seattle, WA, Morgan Kaufmann.

Platt, J. (1998). "How to implement SVMs." *IEEE Intelligent Systems* (July/August): 26-28.

Platt, J. C. (1999). Fast training of support vector machines using sequential minimal optimization. *Advances in Kernel Methods*. B. Scholkopf, C. J. C. Burges and A. J. Smola. Cambridge, Massachusetts, The MIT Press: 185-208.

Schapire, R. E. and Y. Singer (1999). "Improved boosting algorithms using confidence-rated predictions." *Machine Learning* 37(3): 297-336.

Tanabe, L. and Wilbur, W. J. (2002) "Tagging gene and protein names in biomedical text." *Bioinformatics* 18: 1124-32.

Wilbur, W. J. (2002) "A thematic analysis of the AIDS literature." *Pac Symp Biocomput.*: 386-97.