Concept recognition and the TREC Genomics tasks

J. Gregory Caporaso William A. Baumgartner, Jr. K. Bretonnel Cohen
Helen L. Johnson Jesse Paquette Lawrence Hunter

larry.hunter@uchsc.edu
Center for Computational Pharmacology
University of Colorado Health Sciences Center

Abstract

We applied concept recognition techniques to the Genomics track primary and secondary tasks. For the primary task, we developed a foundational information retrieval system which incorporated Entrez Gene entries and UMLS concepts for query expansion via phrasal and term boosting representations of synonyms. For the secondary task, we evaluated three conceptual features—mouse strain names, indexed MeSH terms, and normalized citations—in addition to two surface linguistic features—BOW and bigrams. Our final feature set yielded consistently high F-measures.

Introduction

The Center for Computational Pharmacology at the University of Colorado Health Sciences Center participated in the *ad hoc* retrieval and document classification tasks of the Genomics track. Our approaches to both tasks were based on applying concept recognition techniques wherever possible. For some elements of the secondary task, this yielded high F-measures—above the median in all cases, and the highest F-measure reported for the E task.

Early work in biomedical information extraction systems relied on either direct pattern matching (e.g. [8], [23]) or on machine learning techniques applied directly to the input text (e.g. [12]). However, there is a growing awareness ([16], [11]) that text data mining in the molecular biology domain can benefit from transformations that map from text to more semantically oriented intermediates. In fact, recognizing semantic concepts when they appear in natural language texts has become a major focus of current work in biomedical NLP. Examples of this phenomenon include the entity normalization task in BioCreative ([18]) and the protein-to-Gene-Ontology task in BioCreative ([9], [14]). The goal of concept recognition is to move beyond the named entity recognition task—locating strings in text that refer to some entity—and extend it by mapping the text string to a representation of the entity to which it refers. For example, in the case of BioCreative's entity normalization task, the entities were Entrez Gene entries, and the required mapping specified the Entrez Gene ID. In the case of BioCreative's Task 2, there were two types of entities to be recognized—PDB entries, and Gene Ontology terms. Entries had to map to the appropriate PDB database identifier and a specific Gene Ontology concept. Cohen and Hersh suggest that better concept recognition can improve performance on other tasks ([11]:62,67); here we demonstrate increases in F-measure on document classification tasks when concept recognition is incorporated into our system.

The specific conceptual types with which we experimented were UMLS concepts in the case of the primary task, and mouse strain identifiers in the case of the secondary task. We discuss the details of the concept recognition approaches in the Methods sections below.

Ad Hoc Retrieval Task

Methods

Framework Lucene [1] was used as the backbone for our system. Query generation centered around the seven template component classes—*method or protocol, gene, mutant gene, disease, biological process, organ function,* and *biological impact*—found in the five topic templates. A query generation procedure was created for each of the five topic templates. These procedures included query expansion where applicable. Figure 1 gives an overview of our retrieval system.

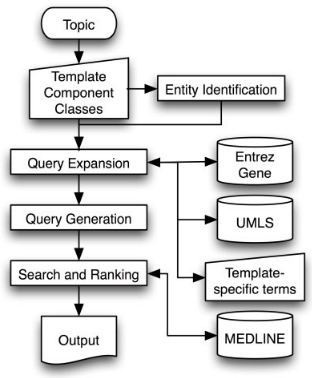

Figure 1: IR System Framework

Entity Identification An entity identification (EI) system developed in-house for the BioCreative competition [20] was used to search for gene names within the text for the *method or protocol* topic.

Query Expansion A variety of query expansion methods were tried in our IR system. They can be loosely classified into three categories—template-dependent, gene-dependent, and disease-dependent. Template-specific terms were added to each query as depicted in Table 1.

Table 1: Template-specific query expansion terms

Template	Template-specific terms
1	method, protocol
2, 3	role
4	interact, promote, suppress, inhibit
5	mutate, mutant, single nucleotide polymorphism, missense, nonsense, insertion, deletion

Query expansion on gene names involved the use of a search index of the Entrez Gene [2] database built using Lucene. This search index encompassed gene name synonyms and associated protein product name synonyms of all Entrez Gene entries. When the EI system found a gene name in the query, we conducted a search for synonyms by querying the Entrez Gene index. We automatically removed these words when they were part of a synonym: *putative, hypothetical, probable, unspecified, unclassified, unspec, disease, nos, disorder, specification,* and *family.*

For topics from the second and fourth templates, disease names were expanded with synonyms found in the UMLS. We found disease synonyms by searching the UMLS MetaThesaurus [4]. We did some hand-filtering of the UMLS synonyms.

We tried two ways of incorporating gene and disease synonyms into the queries, either as phrases or as a set of boosted individual words. Boosting of the individual terms reflected the frequency with which they were found in the synonym set: if a term was seen more frequently in the set, then it was weighted higher than a term that was observed less frequently.

Query Generation Queries were formulated by concatenating the query expansion output with a pre-processed version of the original topic text. The pre-processing of the original topic text consisted of whitespace tokenization and stop word removal.

Search and Ranking A search index of the supplied MEDLINE collection encompassing the stemmed text from all document titles and abstracts was created using Lucene. Stemming was done with the Snowball implementation [3] of the Porter stemming algorithm [24]. Results were ranked using the Lucene modified *tf*idf* scheme. Terms found in the document title were weighted more heavily than those found in the abstract.

Output We submitted two runs from our IR system, labeled *CCP0* and *CCP1.* Generation of the result sets differed in gene and disease synonym representation and the degree of disease synonym filtering. Result set *CCP0* used the synonym phrases as they were returned from our synonym search engine. Result set *CCP1* used the boosted term representation for both gene and disease synonyms.

Results

Judging from the preliminary results released by TREC, our IR system generally performed below the median. The overall Mean Average Precision (MAP) scores for *CCP0* and *CCP1* were 0.1078 and 0.0554, respectively. A comparison of our precision after ten (P10) documents and precision after one hundred (P100) documents with the other manual/interactive entries is shown in Figure 2.

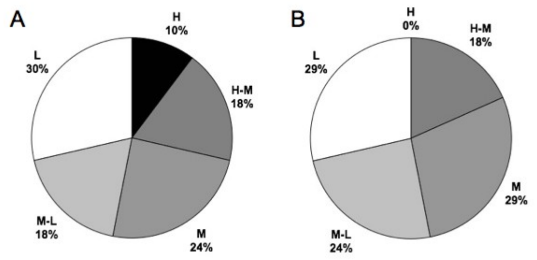

Figure 2: Comparison of the P10 (A) and P100 (B) scores for *CCP0* over all manual/interactive entries. For a given query, our score matched the highest (H, black), fell between the highest and median (H-M, dark gray), matched the median (M, gray), fell between the median and lowest (M-L, light gray), or matched the lowest (L, white) score.

Document Classification Task

Methods

We developed five different feature extractors, described below. We then did feature selection on a per-task basis to select the features with the largest information gain. High-information-gain features were used to train document classifiers. We identified the top-performing classifiers using 10-fold cross-validation, optimizing for normalized utility score. All feature selection and document classification algorithms were implemented within the Weka 3 data mining software package [26].

Feature Extraction

The features were Bag-of-Words (BoW), bigrams, citations, MeSH headings, and strain names. For some features (detailed below) we experimented with extracting features only from specific document sections, and we compared binary and weighted values for some features. Comparing the performance of classification using differing features and groups of features led us to include some of these and exclude others in our final submissions.

Our final classifiers were trained on a combination of bigrams extracted from article titles and captions, MeSH terms associated with documents, and mouse strains identified in the article body.

Bag-of-Words Extractor During construction of the BoW, stop words from the 133-word NCBI list [5] were omitted. Remaining words were stemmed with a Python implementation of the Porter Stemming Algorithm [6]. The text input to the BoW extractor was varied between article bodies, abstracts only, and figure captions only. Feature values were varied between word frequency[1] and binary presence vs. absence.

Bigram Extractor The bigram feature extractor retrieves all of the two-word sequences from a block of text. We stemmed bigrams before counting, so each bigram is actually a two-stem sequence. Bigrams containing stop words were discarded. For example, the filtered bigrams contained within the first clause of this self-referential sentence are: ‘filt bigram’, ‘bigram contain’, ‘first claus’, and ‘self-referenti sentenc’. The text input to the bigram extractor was varied between article bodies, abstracts only, figure captions only, article titles only, and figure captions plus article titles. Feature values were tested as bi-

[1]For a specific feature type, feature frequency was defined as: (Number of times a feature occurs in document) / (Total number of features in document)

gram frequencies and binary values.

Citation Extractor For each citation in an article, we created an identifier based on year of publication, volume number of journal, and first page of article. These identifiers normalized minor differences in author names (*Smith, V.* vs. *Smith, V.M.* vs. *Smith, VM*), journal abbreviations (*J. Mol. Bio.* vs. *Jour. Mol. Bio.* vs. *JMB*), etc. The input to the citation extractor was article bibliographies.

MeSH Term Extractor Since indexed MeSH terms are not listed within the SGML file supplied for each article, the MeSH term extractor is given the PubMed identifier (PMID) for an article as input. The associated MeSH terms are then retrieved from our local installation of MEDLINE.

Strain Name Extractor The strain name feature extractor retrieves mentions of mouse strain names from input text. The strain names searched for are obtained from MGI's 'Official Strain Nomenclature' [7]. Matched terms from this list are mapped onto corresponding MGI identifiers. Feature values were varied between term frequencies and binary values, and input text was varied between full text of articles, and Methods sections[2] only.

Feature Selection

To reduce the dimensionality of our feature sets, we experimented with two feature selection algorithms: Information Gain (IG) and ReliefF. A threshold was set to determine the size of the reduced feature set. For IG, the threshold defines a minimum information gain. For ReliefF, the threshold defines a maximum number of features. In addition to greatly reducing training and testing time, feature selection increased classifier performance in all cases (see *Results*).

To determine the optimal feature set for training classifiers, we varied the feature selection algorithm and the selection threshold. Selection thresholds were varied between 1E-7 and 1E-1 for IG, and between 20,000 and 2,000 features for ReliefF.

[2]We defined 'Methods sections' as any section having a section title of 'Materials', 'Methods', 'Materials and Methods', or 'Experimental Procedures'.

Document Classification

We experimented with Naive Bayes (NB) and Support Vector Machine (SVM) classifiers. The NB classifier was used with kernel estimation activated. SVM classifiers employed the RBF kernel. To find the best performing SVM classifier, a grid search was done on the RBF kernel parameter space, varying gamma between 1E-15 and 1E6, and varying complexity factor between 1E-3 and 1E15 [19].

Classifier Selection

Parameters were selected based on the normalized utility from 10-fold cross validation analyses of training data. We submitted twelve runs, three per task, generated with twelve different classifiers.

Results

Comparison of precision, recall, F-measure, and normalized utility values obtained from 10-fold cross-validation on the training data using different combinations of feature types, feature selection techniques, and classification algorithms led us to select what we expect to be our top-performing classifiers.

Comparison of Feature Types We compared the contribution of the various features using NB classifiers, after IG feature selection with a threshold of 1E-3. For the bigram, BoW, and strain name extractors, the input text was varied across different sections of the articles.

Figure 3a shows that classifiers built using BoW representations of the text performed poorly; the highest normalized utility achieved was 0.144. As a consequence, we did not incorporate these features in our master feature set. The section of the article for which BoW representations were generated seemed to matter little in overall performance.

The bigram representations of the text yielded significantly better performing classifiers than the BoW representations (Figure 3b). We contrasted bigram extraction from article abstracts, titles, captions, and bodies. Normalized utility varied highly depending on the section from which the bigrams were extracted. The best performance was achieved with bigrams extracted from the article bodies (0.916 for task T), but due to an oversight, in our master feature set we used the next-best performer, which was bigrams from captions.

The citation feature generally achieved low normalized utility scores (Figure 3c). The highest normalized utility achieved, 0.233, was for the A task. However, the precision achieved in each of the tasks with these classifiers was very high. The lowest precision achieved was 0.800 (also for the A task), and a precision of 1.000 was achieved for the E and T tasks. Citation features were not included in the master feature set (see below).

The MeSH term feature yielded good classifiers (Figure 3d). The highest normalized utility score (0.820) was achieved for the A task with this feature set. For all other tasks, normalized utility scores were close to, although lower than, those achieved with the classifiers built from bigram feature sets. We included MeSH terms in our master feature set.

The strain name feature achieved similar results to the citation feature (Figure 3e). The strain name extractor was run over full article texts, and over Methods sections only. Better classifier performance was observed for all tasks when strain names were extracted from the full article bodies rather than Methods sections only. Strain name features were included in the master feature set (see below for explanation).

The combination of bigram and MeSH term features into one set constituted our 'base feature set.' To evaluate whether to include the citation and strain name features, we compared their performances when added to the base feature set. (Figure 3f)

When our citation feature set was combined with our base feature set and (following IG feature selection) used to train NB classifiers, we observed very slight increases in performance for the A and G tasks, but large decreases in performance in the E and T tasks. We therefore excluded citation features from our master feature set[3].

When classifiers were trained with our base feature sets plus strain name features, we observed increased classifier performance for all tasks[4] when compared to our base feature set alone. This result informed our decision to combine our base feature set with the strain name feature for our master feature set.

Feature Selection To decide on a good feature selection algorithm and threshold, we experimented with Information Gain (IG) and ReliefF feature selection algorithms and varied the size of the feature sets generated with each.

To determine the optimal selection threshold for the IG algorithm, we filtered the master feature set on a per-task basis with varied selection thresholds. Each filtered feature set was then used to train a NB and an SVM classifier. Performance of each was judged based on normalized utility from 10-fold cross validation on the training data. As illustrated in Figure (4), the feature set sizes which yielded the best document classifiers were achieved with selection thresholds of 1E-3 or 1E-4, depending on the task and classifier type. The ReliefF algorithm generated feature sets whose classifiers performed very poorly in comparison to those generated from the IG-selected feature sets. Since we had good results with IG for feature selection, we set our selection thresholds for ReliefF to generate feature sets of comparable size to those generated with our top performing IG feature sets. For each of the tasks, we saw normalized utilities under 0.288 (and often below 0.1) with selection thresholds set between 2,000 and 20,000. We therefore focused the rest of our efforts on using IG with selection thresholds of 1E-3 and 1E-4.

The higher selection threshold (1E-3) yields feature sets with fewer features than the lower selection threshold (1E-4). We therefore refer to the feature sets generated with selection threshold of 1E-3 as the smaller feature sets, and those generated with selection threshold of 1E-4 as the larger feature sets.

[3]It is interesting to note that in all tasks, inclusion of the citation data improved precision—quite significantly, in some cases. For example, in the T task, our base feature set led to a precision of 0.711; when citation data was included, the precision went up to 1.000 while maintaining a recall of 0.583.

[4]For the G task, although no increase in normalized utility was observed, a slight increase in recall, at the expense of a slight decrease in precision, was observed. Since the normalized utility score is more related to recall than precision, we considered this to be an increase in performance that might translate to a higher normalized utility when run on test data.

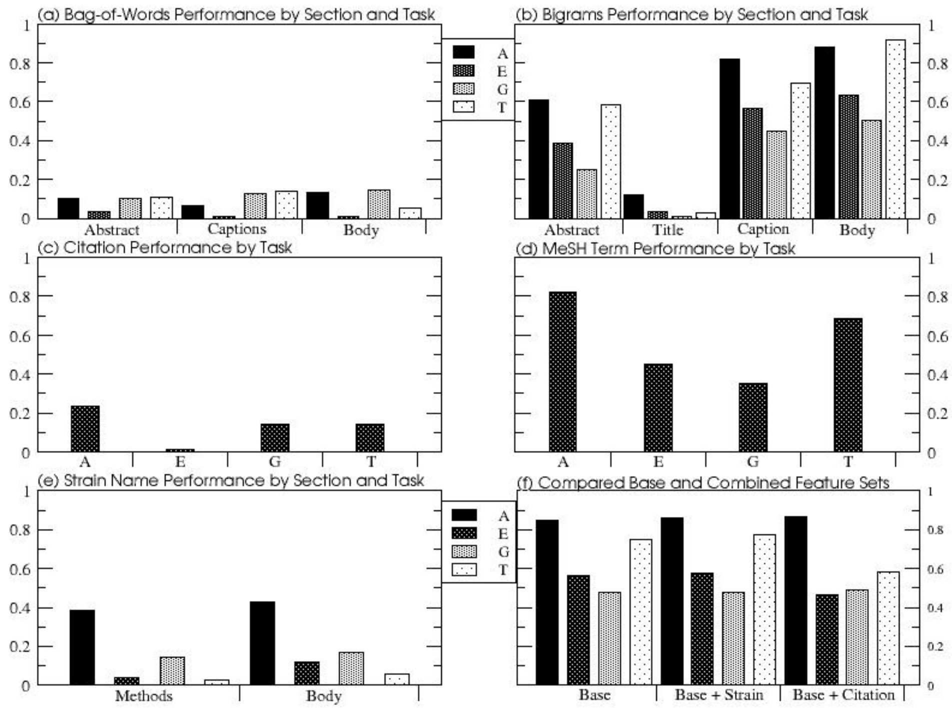

Figure 3: **Comparison of performance of document classifiers trained on varied feature sets.** (a-e) For each feature extractor, classifiers trained for each task are compared on normalized utility from 10-fold cross validation of training data (y-axis). Where applicable, (a,b,e), the article section provided to the information extractor is varied. (f) The base feature set was used to generate a classifier for each task, and the normalized utility achieved with these classifiers is compared with those achieved by training classifiers with the base feature set combined with the strain name feature set or the citation feature set.

Table 2 presents the ten feature values with the highest information gain for each task. The MeSH terms and mouse strains are by definition conceptual. Many of the lexical features, ie. stemmed bigrams, are suggestive of concepts as well. Recognition of these concepts from additional ontologies and semantic classes might therefore improve our classifier performance even further. For example, an 'experimental technique' ontology might allow mapping *saggit[al] section* and *transvers[e] section* to a single *sectional anatomy* concept. This would yield a feature value with higher information gain (as well as reducing the size of the feature space overall).

Document Classifiers We were interested in experimenting with NB and SVM classifiers. Last year it was reported [10] that SVMs with a linear kernel were outperformed by NB classifiers. Since the SVM with the RBF kernel (SVM/RBF) is typically held to outperform SVM with the linear kernel, we were curious to see how SVM/RBF would compare to NB.

In building each of these classifiers, we began with our master feature set, and performed IG feature selection twice per task to generate two sizes of feature sets for each. This yielded a total of eight feature sets. (Note that the selected feature sets will be different

Table 2: **Feature values with highest information gain by task.** Values beginning with 'MGI:' are mouse strain names, normalized to their MGI identifiers. Italicised values are MeSH terms, and the remaining values are bigrams.

A	E	G	T
Mice, Knockout	mous embryo	*Mice*	*Mice*
Mice	*Mice*	*Mice, Knockout*	*Mice Knockout*
MGI:2160041	situ hybrid	MGI:2160041	MGI:2160041
MGI:2160085	neural tube	*Animals*	MGI:2160085
southern blot	*Gene Expression Regulation, Developmental*	MGI:2160085	southern blot
target vector	yolk sac	mous tissu	tumor incid
Animals	embryo b	southern blot	histolog analys
mutant mice	sagitt section	MGI:2159769	apc mice
Mice, Inbred C57BL	transvers section	MGI:2161069	*Mice, Transgenic*
wild-typ mice	branchial arch	b southern	*Intestinal Neoplasms*

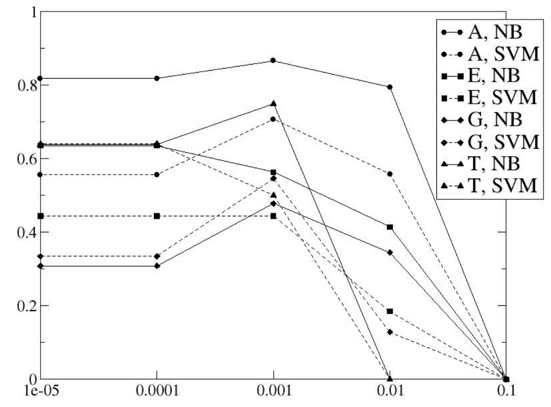

Figure 4: **Effect of feature selection threshold on classifier performance.** The varied Information Gain selection thresholds (x-axis) are plotted versus their effect on normalized utility (y-axis) in 10-fold cross validation on the training data. NB classifiers are represented with solid lines, SVM classifiers with dashed lines.

for each task since different features will be important in discriminating between positive and negative examples of that category.)

To determine the optimal values of the SVM/RBF parameters, we varied the gamma/complexity factor (G/CF) parameters as suggested by [19]. A coarse grid search was performed for the A task using the smaller feature set (Figure 5), and regions of G/CF space which produced optimal classifiers were then investigated with a finer grid search on all four tasks. When SVM/RBF classifiers were compared across the different tasks and parameters, we found that classifiers performed best with a gamma value of 2.5E-2 and a complexity factor value of 2.5E3.

We compared our best-performing SVM classifiers to our NB classifiers (Table 3) for both feature set sizes, and found that in almost all cases NB outperformed the SVM classifiers when compared on normalized utility. The one exception was the G task, when the opposite was true. Additionally, in almost all cases, the smaller feature sets produced better

Table 3: **Comparison of Support Vector Machine and Naive Bayes Classifiers.** Precision, Recall, F-measure and Normalized Utility (U-norm) for NB and SVM classifiers trained with two different sized selections of our master feature set for each task. The 'Threshold' column provides the feature selection threshold used in generating the selected feature set for each run.

		Naive Bayes				Support Vector Machine			
Task	Threshold	Precision	Recall	F-measure	U-norm	Precision	Recall	F-measure	U-norm
A	1.0E-3	0.633	0.893	0.741	0.863	0.888	0.731	0.802	0.725
A	1.0E-4	0.729	0.837	0.780	0.819	0.913	0.559	0.694	0.556
E	1.0E-3	0.681	0.580	0.627	0.576	0.973	0.444	0.610	0.444
E	1.0E-4	0.634	0.642	0.638	0.636	0.923	0.444	0.600	0.444
G	1.0E-3	0.539	0.519	0.529	0.479	0.797	0.593	0.680	0.579
G	1.0E-4	0.725	0.297	0.421	0.286	0.939	0.335	0.494	0.334
T	1.0E-3	0.651	0.778	0.709	0.776	1.000	0.472	0.642	0.472
T	1.0E-4	0.870	0.556	0.678	0.555	1.000	0.639	0,780	0.639

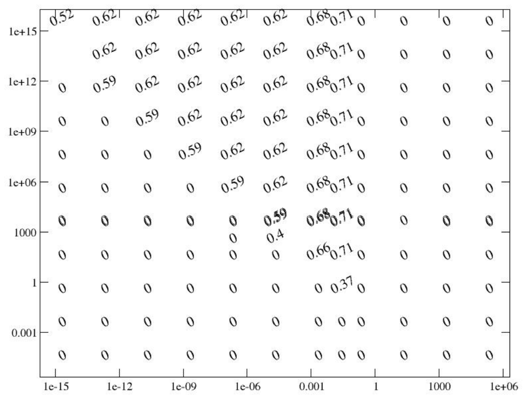

Figure 5: **Grid search of gamma and complexity factor space for SVM/RBF classifier.** For varied values of Gamma (x-axis) and Complexity Factor (y-axis), the normalized utility achieved from 10-fold cross-validation on training data is plotted for the A task. Similar results were observed for the E, G, and T tasks.

classifiers than the larger feature sets[5].

We submitted the maximum of twelve runs (three per task). Our master feature set consisted of bigrams extracted from article titles and figure captions[6], MeSH terms, and strain names extracted from full article bodies. We used IG to construct two filtered feature sets per task, using cut-off thresholds of 1E-3 and 1E-4. We built two sets of NB classifiers (one set per feature selection threshold), and one set of classifiers using SVMs (generated with the smaller feature sets and G and CF values of 2.5E-2 and 2.5E3 respectively) per task.

Performance on the test data Our top-performing classifiers were the NB classifiers with the smaller feature sets for the A, G, and T task, and the NB classifier with the larger feature sets for the E task. Our NB classifiers outperformed our SVM classifiers. Our best normalized utility score, 0.7215, was

[5]It should be noted here that the observed result of NB classifiers outperforming SVM classifiers is only valid when compared on the basis of normalized utility. SVM classifiers always achieved higher precision than NB classifiers, and NB classifiers almost always achieved higher recall. An SVM classifier achieved the top F-measure for three out of the four tasks.

[6]We found that including article title bigrams with article caption bigrams provided a slight but consistent increase in classifier performance, and thus decided to include them in our master feature sets.

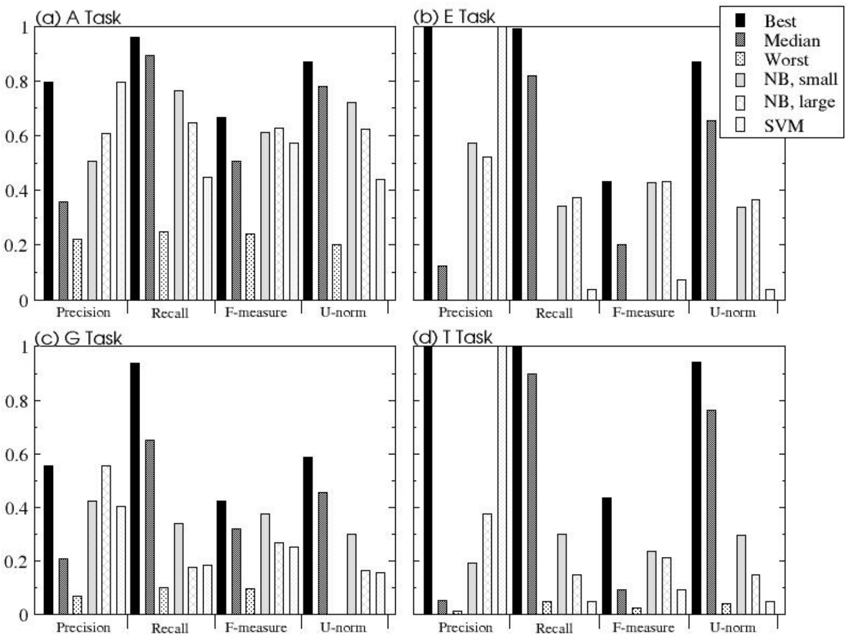

Figure 6: **Comparison of our results with the Best, Median, and Worst overall results.** For each task (a-d) precision, recall, F-measure, and normalized utility achieved by each of our classifiers are compared to the best, median, and worst scores on the test data. For each performance metric, from left to right, the bars illustrate: overall best, overall median, overall worst, our NB classifier with the smaller feature set, our NB classifier with the larger feature set, and our SVM classifier.

achieved in the A task with an NB classifier. With respect to normalized utility, our scores fell below the median for all tasks, but examination of other metrics reveals some of the strengths of our system. For example, we achieved the maximum precision for each task. Our F-measures were generally among the best, and for the E task we achieved the maximum. However, our recall scores were low, leading to low normalized utilities (Figure 6).

Discussion

The reported performance of concept recognition in the TREC Genomics track has varied in interesting ways. In the first year of the track, the track director found that searching in the MeSH and substance name fields, along with filtering for species, accounted for the best performance ([17]:19). In the second year of the track, the track director noted the opposite: "Approaches that attempted to map to controlled vocabulary terms did not fare as well" ([15]:137). (The systems to which they refer are described in [25], [22] and [13].) The second-year observation seems to contradict the idea that concept-based techniques are helpful in this domain, so we examine the specific approaches more closely here. There are at least three distinct explanations for the observation, all of which are consistent with our hypothesis. One possibility is that these approaches may simply have made a poor choice of concepts. Another is that the systems may have done a poor job of concept recognition. Finally, [15] may be overstating the role that controlled vocabularies played in the poorly performing systems.

It is difficult to evaluate the question of the role that the choice of a specific controlled vocabulary played in performance for these systems; they all used the same set of concepts, so we cannot contrast its performance contribution with that of some other set of concepts. Nonetheless, it seems prima facie odd to suggest that the choice of concepts was at fault, since the shared choice for concepts was MeSH, which was in fact designed precisely to facilitate information retrieval. However, [21] review a number of studies that suggest problems with the use of MeSH in IR tasks. In our own work reported here, we had considerable difficulty using UMLS as a concept source in the primary task.

Another possible cause of [15]'s observation is that the three systems might not have done well at concept recognition. [13] note problems with recognition of MeSH terms related to synonymy relations. Neither [22] nor [25] describe their method for matching MeSH concepts, so presumably it was not a focus of their work. In our own work, we had much greater success in the case where the concept recognition task was easier, i.e. strain name identification for the secondary task, than in the case where it was more difficult, i.e. using the UMLS in the primary task.

Finally, we suggest that [15] do, in fact, overstate the role that concept recognition played in these systems. In the case of [22], MeSH terms were only one of several features. In the case of [25], they were used only as a source of synonyms, and there is no data available on how often these were even applied.

In summary, our work here suggests a role for concept recognition in tasks of uncontested relevance to biologists. It seems likely that this role will become even stronger as concept recognition improves.

References

[1] Lucene. `http://lucene.apache.org`.

[2] NCBI, Entrez Gene. `http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=gene`.

[3] Snowball: A language for stemming algorithms. `http://snowball.tartarus.org/texts/introduction.html`.

[4] UMLS: Metathesaurus. `http://www.nlm.nih.gov/research/umls/meta2.html`.

[5] NCBI Stop Words, 2005. `http://www.ncbi.nlm.nih.gov/`.

[6] Python porter stemming implementation, 2005. `http://tartarus.org/~martin/PorterStemmer`.

[7] MGI official strain nomenclature, August,2005. ftp://ftp.informatics.jax.org/pub/reports/MGI_Strain.rpt.

[8] Christian Blaschke, Miguel A. Andrade, Christos Ouzounis, and Alfonso Valencia. Automatic extraction of biological information from scientific text: protein-protein interactions. *ISMB 1999*, pages 60–67, 1999.

[9] Christian Blaschke, Eduardo Andres Leon, Martin Krallinger, and Alfonso Valencia. Evaluation of BioCreative assessment of task 2. *BMC Bioinformatics*, 6 Suppl 1:S16, 2005.

[10] A. M. Cohen, R. T. Bhupatiraju, and W. R. Hersh. Feature generation, feature selection, classifiers, and conceptual drift for biomedical document triage. In *The thirteenth Text Retrieval Conference, TREC 2004*. National Institute of Standards and Technology, 2004.

[11] Aaron M. Cohen and William Hersh. A survey of current work in biomedical text mining. *Briefings in Bioinformatics*, 6(1):57–71, 2005.

[12] Mark Craven and Johan Kumlien. Constructing biological knowledge bases by extracting information from text sources. *ISMB 1999*, 1999.

[13] Alan R. Aronson et al. Knowledge-intensive and statistical approaches to the retrieval and annotation of genomics MEDLINE citations. In *The thirteenth Text Retrieval Conference, TREC 2004*. National Institute of Standards and Technology, 2004.

[14] Evelyn B. Camon et al. An evaluation of GO annotation retrieval for BioCreative and GOA. *BMC Bioinformatics*, 6 Suppl 1:S17, 2005.

[15] William Hersh et al. TREC Genomics track overview. In *The Thirteenth Text Retrieval Conference, TREC 2004*, pages 132–150. National Institute of Standards and Technology, 2004.

[16] Carol Friedman, Pauline Kra, Hong Yu, Michael Krauthammer, and Andrey Rzhetsky. GENIES: a natural-language processing system for the extraction of molecular pathways from journal articles. *Bioinformatics*, 17(Suppl. 1):S74–S82, 2001.

[17] William Hersh and Ravi Teja Bhupatiraju. TREC Genomics track overview. In *The twelfth Text Retrieval Conference, TREC 2003*, pages 14–23. National Institute of Standards and Technology, 2003.

[18] Lynette Hirschman, Marc Colosimo, Alexander Morgan, and Alexander Yeh. Overview of BioCreative Task 1B: normalized gene lists. *BMC Bioinformatics*, 6 Suppl 1:S11, 2005.

[19] Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin. A practical guide to support vector classification.

[20] S. Kinoshita, K. B. Cohen, P. V. Ogren, and L. Hunter. BioCreAtIvE Task1A: entity identification with a stochastic tagger. *BMC Bioinformatics*, 6 Suppl 1:S4, 2005.

[21] Ronald N. Kostoff, Joel A. Block, Jesse A. Stump, and Kirstin M. Pfeil. Information content in Medline record fields. *International Journal of Medical Informatics*, 73:515–527, 2004.

[22] PI Nakov, AS Schwartz, E Stoica, and MA Hearst. BioText Team experiments for the TREC 2004 Genomics track. In *The thirteenth Text Retrieval Conference, TREC 2004*. National Institute of Standards and Technology, 2004.

[23] T. Ono, H. Hishigaki, A. Tanigami, and T. Takagi. Automated extraction of information on protein-protein interactions from the biological literature. *Bioinformatics*, 17(2):60–67, 2001.

[24] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[25] Kazuhiro Seki, James C. Costello, Vasanth R. Singan, and Javed Mostafa. TREC 2004 Genomics track experiments at UIB. In *The thirteenth Text Retrieval Conference, TREC 2004*. National Institute of Standards and Technology, 2004.

[26] Ian H. Witten and Eibe Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, 2nd edition, 2005.