Three Approaches to GO-Tagging Biomedical Abstracts

Neil Davis, Henk Harkema,
Rob Gaizauskas, Yikun Guo
initial.surname@dcs.shef.ac.uk
Department Of Computer Science
University Of Sheffield

Moustafa Ghanem,
Tom Barnwell, Yike Guo
initial.surname@imperial.ac.uk
Department of Computing
Imperial College

Jon Ratcliffe
jratcliffe@inforsense.com
InforSense Ltd

Abstract

In this paper we explore three approaches to assigning Gene Ontology semantic classifications to abstracts from the PubMed database: lexical lookup, information retrieval and machine learning. To evaluate the approaches we use two “gold” standards derived from the yeast genome database (SGD). While evaluation provides insights into the three approaches, it also reveals the difficulties in constructing a suitable gold standard for this task.

1 Introduction

Text mining may be described as the process of revealing information, regularities, patterns or trends in textual data. As such it draws upon a variety of fields, including information extraction (IE), information retrieval (IR), natural language processing (NLP), knowledge discovery from databases (KDD) and traditional data mining (DM) (Hearst, 1999). Text mining is of particular interest to the biological scientist because of the on-going explosive growth of the biomedical literature. With literature being published at ever increasing rates, it is becoming more and more difficult for the researcher to keep abreast of developments in his own area or to make connections with related areas. Text mining attempts to address these problems in various ways. For example, through *extractive* processes, facts or terms may be extracted from papers and made available for searching or automatic linking; through *structuring* processes, papers may be automatically grouped or organised based on content, facilitating conceptual access to large numbers of scientific papers.

We consider this latter approach here. The current nomenclature for genes and proteins is diverse and frequently almost ad hoc in its nature, making it very difficult to relate structurally or functionally similar genes and proteins across different species. In order to begin to integrate the knowledge about these shared biological entities, a common descriptive framework is required and this is the challenge that the Gene Ontology (GO) seeks to undertake (Gene Ontology Consortium, 2000). GO itself consists of three separate ontologies, each describing a particular facet of molecular biology: molecular functionality, biological processes and cellular components. The annotation of the biomedical literature with terms from GO could provide a semantic overview of the literature and a way of organising the knowledge it contains. However, given the rapid growth in the biomedical literature, manual annotation with GO codes is not feasible. Some automatic method of annotation would be of significant interest, both for model organism database curators who are increasingly using GO as an important part of their annotations, and also for the biological scientist for whom GO annotations of papers could be used to deliver conceptual level search and browsing tools. We address the task of annotating papers from the biomedical literature with Gene Ontology in this paper.

2 The Task and Related Work

2.1 GO and the Use of GO

GO consists of almost 20,000 terms organised into three separate ontologies.[1] Each term consists of a name, a unique identifier (GO code), an indication of which ontology it belongs to, an optional list of synonyms (of various types) and a textual gloss or definition. For example:

Term name:	*isotropic cell growth*
Accession:	*GO:0051210*
Ontology:	*biological_process*
Synonyms:	*related: uniform cell growth*

[1] See http://www.geneontology.org/.

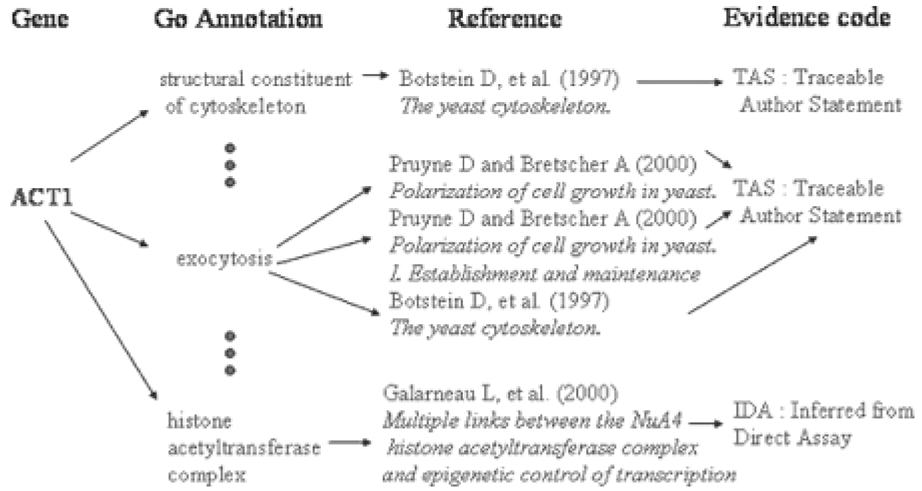
Gene
Go Annotation
Reference
Evidence code
ACT1
structural constituent of cytoskeleton
Botstein D, et al (1997) The yeast cytoskeleton.
TAS : Traceable Author Statement
exocytosis
Pruyne D and Bretscher A (2000) Polarization of cell growth in yeast.
Pruyne D and Bretscher A (2000) Polarization of cell growth in yeast. I. Establishment and maintenance
Botstein D, et al (1997) The yeast cytoskeleton.
TAS : Traceable Author Statement
histone acetyltransferase complex
Galarneau L, et al. (2000) Multiple links between the NuA4 histone acetyltransferase complex and epigenetic control of transcription
IDA : Inferred from Direct Assay

Figure 1: Gene, GO annotation, Reference and Evidence Code relationships with SGD

> *Definition: The process by which a cell irreversibly increases in size uniformly in all directions. In general, a rounded cell morphology reflects isotropic cell growth.*

These terms are organised into a directed acyclic graph, whose edges indicate the semantic relation IS-A-TYPE-OF, so that for all terms, if a child term describes a gene product, then all its parent terms must also apply to that gene product. Note that GO does not list or reference gene products directly; rather it provides a controlled vocabulary for describing them.

One common use for GO terms is to associate them with genes or gene product entries in model organism or protein databases. For example, databases such as the Drosophila Genome Database (FlyBase)[2], the Saccharomyces Genome Database (SGD)[3] and the Mouse Genome Database (MGD)[4] associate one or more GO annotations with each gene in the database. Each GO annotation has one or more references to the literature,providing support for the annotator's decision to link this GO code to this gene. Each reference given in support of a GO assignment also has an evidence code associated with it, which provides a characterisation of the type of evidence the reference gives for this GO code assignment. These complex relations are illustrated in Figure 1.

This usage of GO codes is prototypical. GO was not designed, first and foremost, as an indexing scheme for texts (as were the Medical Subject Headings or MeSH). However, this does not mean that doing so is not a worthwhile, if difficult task Associating one or more GO codes with a text if the text is "about" the function/process/component identified by the code would provide a domain-specific approach to organising the text collection. One issue is whether to assign GO terms at the level of the whole document or to assign them to textual sub-units (paragraphs, sentences, gene names) to which they may more specifically apply. We have focused on the task of assigning codes to the biomedical (PubMed) abstracts, as this task is challenging and useful in its own right.

2.2 Related work

Raychaudhuri et al. (2002) address the task of associating GO codes with genes by first associating GO codes with documents and then assigning a specific GO code to a gene if a sufficient number of documents mentioning the gene have had the GO code associated with them. They treat GO code assignment to documents as a document classification task and evaluate maximum entropy, Naive Bayes and nearest-neighbours approaches. To evaluate they assemble a corpus of roughly 20,000 PubMed abstracts associated with one or more of 21 GO terms/categories. This is done by searching PubMed for each GO term's corresponding MeSH heading and title words. On the document-level task, the maximum entropy approach proves best, obtaining 72.8% classification accuracy over the 21 categories.

GO-KDS (Smith and Cleary, 2003) uses a Naive Bayes-like machine learning algorithm called the

[2] See `http://flybase.bio.indiana.edu/`.

[3] See `http://www.yeastgenome.org/`.

[4] See `http://www.informatics.jax.org/`.

Weighted Confidence Learner (WCL), Bayes, developed specifically for the task of annotating PubMed articles with GO terms. Taking only words as document features, GO-KDS uses a scoring function to rank each document within each GO category. Category membership is then determined by applying a score threshold for each category. GO-KDS was trained using gene/protein databases that employ GO terms and also have references to MEDLINE papers (initially some 26,500 documents with 3700 GO terms were used). It was evaluated on approximately the same data as Raychaudhuri et al.'s (2002) approach and attained a classification accuracy of 70.5%. GO-KDS has the advantage of being extremely efficient, an important consideration given its intended use in a commercial product suite.[5]

GoPubMed is another tool designed to annotate PubMed abstracts with GO terms (Doms and Schroeder, 2005). A local sequence alignment algorithm is used based on a weighted term matching between GO terms and strings in the abstracts that permits words to be dropped and ranks certain words more highly than others. No evaluation of this work is reported.

Kiritchenko et al. (2005) present a method for assigning GO codes to biomedical texts. They treat GO term assignment as a text categorisation problem and approach it using the AdaBoost.MH algorithm. The novelty in their work lies in treating the problem as a *hierarchical* text categorisation task and in introducing a new evaluation measure for hierarchical categorisation tasks that gives credit to partially correct classifications. Several evaluation results are provided but it is not obvious whether the same algorithm is being compared using different evaluation measures, or different algorithms using the same measure. Also, with regard to the creation of a gold standard for evaluation purposes, the authors seem unconcerned about the completeness issues that are raised below in section 3.

The BioCreAtIvE challenge (Hirschman et al., 2005) was held to provide a set of common evaluation tasks to assess the state of the art for text mining applied to biological problems. Task 2 focused on the automatic assignment of GO to human proteins. In subtask 2.1 participants were given an article, a protein and a GO code, where the article justifies the assignment of the GO code to the protein, and required to find evidence text in the article supporting the assignment. In subtask 2.2 participants were given protein-article pairs plus the number of GO code assignments supported by the article, and required to find the GO code(s) that should be assigned to the protein based on the article. 200 full-text articles manually annotated by curators were used as the test set. Results indicated no systems are ready for practical use as yet.

3 Gold Standards and Data Sets

Two gold standards were created for assessment purposes. The first uses the GO annotations in the Saccharomyces Genome Database(SGD) (Cherry et al., 1997) (later referred to as SGD Abstracts); the second addresses problems of completeness in the first by manually extending it with mentions of GO terms (later referred to as IC Abstracts).

3.1 SGD Abstracts Gold Standard

As described in section 2.1, SGD records for each (yeast) gene a list of GO terms manually curated to that gene. Each such assignment of a GO term to a gene has associated with it one or more literature references that provide support for the assignment.

The annotated gene list from SGD is turned into a Gold Standard for our GO tagging task by extracting all references to PubMed articles from the list and assigning to each article the GO terms in the context of which the article was referenced in the list. The resulting Gold Standard contains 4922 PMIDS (PubMed identifiers) and 2455 GO terms, forming 10485 PMID-GO term pairs.

The advantage of creating a Gold Standard in this way is that the information is readily available from SGD; no further annotation work is required and the resulting Gold Standard can be used as a common benchmark since it is available to all. Moreover, other model organism databases can be used to assemble additional Gold Standards in the same way.

However, a major disadvantage of using SGD to create a Gold Standard for GO tagging is that the list of gene-GO term assignments is incomplete for our particular task. First of all, for a given assignment, the list does not necessarily contain *all* papers supporting that assignment. Secondly, since the SGD curation process concentrates on citing evidence for assigning GO terms to genes, the list may miss GO terms which are not directly involved in assignments, but which can neverthe-

[5] See `http://www.go-kds.com/`

less be legitimately attached to a paper according to our task description (see section 2.1). As a consequence, the Gold Standard will be incomplete: for any given paper it will generally not provide the complete set of GO terms that should be assigned to the paper. This means that the Gold Standard can only be used as a weak measure for recall (a system with higher recall is a better system, but 100% recall does not mean perfection). Any precision figures computed from this Gold Standard cannot be interpreted in a meaningful way.

A further disadvantage of the SGD-based Gold Standard is that the information in SGD is derived from full papers, whereas our system has access to abstracts of papers only. Hence, there will be GO terms in the Gold Standard which can only be found by reading the full paper. We cannot reasonably expect our system to find these terms. This feature of the SGD Gold Standard affects the maximum recall obtainable by our GO Tagger.

3.2 IC Abstracts Gold Standard

To address these problems, a second gold standard was created. First, the original SGD Gold Standard was filtered to keep only the abstracts in which a reference to all the GO terms assigned to that paper could be found in either the title or abstract text. This aimed to resolve the issue of assignments being made on the basis of the full paper rather than just the abstract.

Then an attempt was made to find new GO terms referred to in the title or abstract text but not included in the original SGD Gold Standard. To attempt this manually is an extremely challenging task, not least because it requires a thorough knowledge of GO. Therefore a semi-automated process was employed using a fuzzy lookup method to identify candidate references to GO terms and then manually post-process these candidate references to filter out those considered invalid and to select the correct GO term when references in the text overlapped. This resulted in a gold standard with 785 PMIDs and 1006 GO terms forming 5170 PMID-GO term pairs.

The annotations for the IC abstracts are more complete, though still not entirely so; they still contain only those GO terms which are directly mentioned in the text rather than being semantically inferred. A second problem with the IC abstracts set is that it was semi-automatically generated using a lexical lookup system. This means that, apart from the initial GO terms annotated by the SGD, all the GO terms annotated will be present as actual mentions within the text and so amenable to extraction by the same lexical lookup approach that was used to find them.

4 Approaches to the Task

We have tested three approaches to the problem of annotating biomedical abstracts with GO codes.

4.1 Lexical lookup Using Termino

The first approach to the GO annotation task is based on lexical lookup. A text is assigned a particular GO identifier if the corresponding name occurs in the text. Texts that contain multiple names are annotated with multiple GO identifiers.

The lexical lookup method is simple and fast. While it suffers from the obvious drawback that GO names are not selected because they are strings that occur in natural text, but rather because they are appropriate concept names, this approach is a sensible baseline and additionally can supply features to be used in a machine learning approach.

The implementation of the lexical lookup method is based on Termino, a flexible, large-scale terminological resource supporting term processing for text mining applications (Harkema et al., 2004). Termino contains a database holding large numbers of terms and information about these terms, which can imported from existing terminological resources such as UMLS and GO. Efficient recognition of terms in text is achieved through the use of finite state recognisers which are compiled from contents of the database.

For the GO tag application, we harvested all 18270 names from the Gene Ontology (version 120120051108), with their GO identifiers and Ontology attributes and imported these into Termino's database. Synonyms were not included. Names of obsolete GO terms were imported, but flagged, so that they can be excluded from recognition if desired. The term recogniser derived from the GO data is case-insensitive and performs "simple" morphological analysis (e.g., *cells* is recognised as a variant of *cell*, but *mitochondrial*, *mitochondria* is not reduced to *mitochondrion*).

4.2 GO Assignment as an IR Task

Our second approach was to treat GO code assignment as a nearest neighbours task using an Information Retrieval (IR) system to determine proximity. For each GO term, a "GO document" was derived from the GO ontology. Each GO document consists of the GO term name, its definition, and its synonyms. This document collection was indexed using Lucene.[6] A given biomedical abstract is annotated with GO terms by supplying the body of the abstract to Lucene as a query, which will return a rank-ordered set of the most similar GO documents. The top ranked GO documents, whose scores are higher than a given threshold, are used as the GO assignments for the "query" abstract.

In addition to treating each document as "flat", i.e. independent of other GO nodes in the ontology, we also tried a hierarchical variant, in which each GO document also inherits all the GO term names, definitions and synonyms from its parents.

Each GO document, for either variant, goes through a series of standard preprocessing procedures – tokenization, stopword removal, case normalisation, and stemming – and is then indexed by Lucene. The same procedures are applied to the query abstracts from which a query is derived by boolean ORing the words.

Two variants of the GO ontologies were used: the full version, which contains some 18270 nodes, and an abbreviated ontology authored by the SGD curators (Yeast GO slim). Both GO variants were indexed, with and without hierarchical variations, to give four potential search indices.

4.3 GO Assignment Using ML

The third approach explores the possibilities of using machine learning. The task was considered to be one of classification, whereby a system is taught by example how to assign a GO term to a document.

Such a document classification system was built and trained using a corpus of documents with correctly assigned GO terms. Once trained the system was used to predict the GO terms that should be assigned to previously unseen documents.

The Gold Standard data sets (see section 3) were partitioned when using the ML approach to generate a training data set and an evaluation data set. The training data set contained 66% of the documents in the gold standard and the remainder formed the evaluation data set. The partition was performed randomly whilst maintaining the same distribution of GO terms in both sets.

Words and frequent phrases were used as the features on which the classification was based. The classification algorithm uses the per document frequency of these features to learn the GO term class or classes it should assign an abstract to. Words were generated by simple tokenization. Frequent phrases were generated by tokenizing the abstracts, removing stopwords (*a*, *the*, *and* etc.), stemming the remaining words and then identifying frequently occurring sequences of tokens.

The first document classifier tested used a Naive Bayes classification algorithm as implemented in the DiscoveryNet workflow-based knowledge discovery environment (Rowe et al., 2003). The shortcoming of this implementation was that it only assigned one GO term per document, whereas in our Gold Standard data sets, one document may be correctly assigned more than one GO term. This obviously limited recall.

For this reason two further document classification systems were tested. Firstly, the Rainbow text classifier was used. [7] This provides an implementation of the Naive Bayes algorithm using words as features. Secondly the document classifiers within Oracle were used. Oracle Text provides document classification methods based on an SVM or a decision tree algorithm,[8] both of which were tested.

GO includes nearly 20,000 terms and so any data set used to train a classifier is unlikely to include documents assigned to every term. For this reason it was decided to use an abridged version of GO (Generic GO Slim) for which we could get good coverage in a training data set.

5 Results and discussion

5.1 Termino Results

Table 1 summarises the performance of the lexical lookup approach on the GO tagging task. The table contains four sets of results: for each version of the Gold Standard (SGD Abstracts, IC Abstracts), performance figures are given against the full set of GO terms and against GO Slim.[9]

The relatively low recall scores for the SGD abstracts are as expected. This is a result partly of the shortcomings of the SGD Gold Standard (see

[6] See `http://lucene.apache.org`.

[7] See `http://www.cs.cmu.edu/~mccallum/bow/rainbow/`.

[8] See`http://www.oracle.com/technology/products/text/`.

[9] The GO Slim numbers are obtained by mapping the GO terms from the complete GO ontology onto their corresponding terms in GO Slim, both in the Gold Standard and in the results produced by lexical lookup, and comparing the versions of the Gold Standard and the results created in this way.

Gold Standard	GO Version	Recall	Precision	F-measure
SGD Abstract	GO	15.1	5.3	7.8
SGD Abstracts	GO Slim	51.0	29.9	37.7
IC Abstracts	GO	71.9	90.5	80.2
IC Abstracts	GO Slim	79.5	98.5	88.0

Table 1: GO tagging results for lexical lookup approach

		SGD Abstracts					IC Abstracts				
GO Index		1	5	10	20	50	1	5	10	20	50
GO Slim – flat	Recall	17.8	51.5	69.5	82.0	90.2	14.5	42.1	59.5	74.1	85.3
	Precision	34.3	26.2	21.6	16.3	10.2	57.5	44.2	37.6	29.8	19.5
	F	23.5	34.7	32.9	27.1	18.3	23.2	43.1	46.1	42.5	31.7
GO Slim – hierarchical	Recall	19.3	51.0	68.0	80.6	90.3	15.7	43.3	59.5	73.2	85.2
	Precision	31.7	24.8	20.7	15.9	10.0	53.5	43.0	37.0	29.5	19.2
	F	24.0	33.4	31.7	26.5	18.1	24.3	43.1	45.5	42.1	31.4
GO – flat	Recall	5.1	12.3	17.3	23.0	31.7	2.8	6.9	10.0	14.0	20.9
	Precision	11.1	5.3	3.7	2.5	1.4	18.8	9.2	6.7	4.7	2.8
	F	7.0	7.4	6.1	4.5	2.6	4.9	7.8	8.1	7.1	5.0
GO – hierarchical	Recall	3.7	9.2	12.9	17.6	25.7	1.7	3.7	5.5	8.4	13.3
	Precision	7.9	4.0	2.8	1.9	1.1	11.3	5.0	3.7	2.8	1.8
	F	5.0	5.6	4.6	3.4	2.1	2.9	4.3	4.4	4.2	3.2

Table 2: GO tagging results using an IR approach

section 3), partly of a mismatch between nature of task and an approach based on lexical lookup of GO terms (for a GO term to be associated with an abstract it does not necessarily have to appear in the abstract or full paper), and partly of drawbacks inherent to straightforward lexical lookup (this method is not equipped to deal with natural variations of terms, including, for example, permutation and derivation, e.g., *regulation of translation* vs. *regulated translation*, and truncation, *galactokinase activity* vs. *galactokinase*). The jump in recall scores for the SGD abstracts between GO and GO Slim is explained by the fact that it is easier to recognise a GO Slim term than to recognise a GO term: the latter requires recognition of the exact term, whereas the former requires the (exact) recognition of just one of the many terms from GO that are mapped onto the particular GO Slim term.

To the extent that the precision figures computed from the SGD abstracts can be interpreted, we notice that they are lower than expected in the present situation where terms generally do not overlap with common English words. For various reasons, the occurrence of a GO term in an abstract does not mean that this GO term has been assigned to the abstract in the SGD Gold Standard. For example, the term may be too general to be informative, e.g., *cell*, or the recognised term may be part of a larger term (that may or may not have been recognised itself), e.g., *mitochondrion* in *mitochondrion distribution*. For the SGD Gold Standard, precision increases from GO to GO Slim because some recognised terms that are incorrect according to the Gold Standard based on GO will be mapped onto terms that are correct according to the Gold Standard based on GO Slim (while the reverse does not happen). Hence, some false positives turn into true positives.

The recall and precision figures for the Gold Standard based on the IC abstracts look rather better than the SGD abstracts. However, these high scores are mainly an artefact of the way the IC Gold Standard was constructed (see section 3). The differences between the numbers for GO and GO Slim are similar to those observed for the IC Gold Standard, but smaller in magnitude.

5.2 Information Retrieval Results

Table 2 gives the performance figures for the IR approach described in section 4.2 on both the SGD and IC gold standard test sets. Four versions of GO documents, i.e., GO Slim, GO and their hierarchical variants, were used and the table shows the performance figures for the top 1, 5, 10, 20 and 50 GO documents are retrieved for each version. From the table we can see that in general the IR approach performs better on the IC test set than on the SGD test set, due to the relative completeness of the IC gold standard. However, the performance increase is not as significant as with the other two methods described in this paper. This may result

Gold Standard	Classifier	GO Version	R	P	F	Notes
SGD Abstracts	Naive Bayes: words + phrases	GO Slim	15.8	54.6	24.5	1
SGD Abstracts	Naive Bayes: words	GO Slim	17.8	61.7	27.6	1
SGD Abstracts	Naive Bayes: phrases	GO Slim	16.6	57.3	25.7	1
SGD Abstracts	Oracle Text decision tree	GO Slim	36.8	51.6	43.0	2
SGD Abstracts	Oracle Text SVM	GO Slim	17.5	53.4	26.4	2
SGD Abstracts	Rainbow	GO Slim	25.8	55.8	35.3	3
IC Abstracts	Naive Bayes: words + phrases	GO Slim	10.4	73.6	18.2	1
IC Abstracts	Naive Bayes: words	GO Slim	11.6	81.7	20.3	1
IC Abstracts	Naive Bayes: phrases	GO Slim	10.7	75.7	18.7	1
IC Abstracts	Oracle Text decision tree	GO Slim	76.5	83.0	79.6	2

Notes: 1. Classifier can only predict one GO term per abstract.
2. Classifier can predict more than one GO term per abstract.
3. Classification confidence parameter has been optimised.

Table 3: GO tagging results using a machine learning approach

from how IC gold standard is generated, which favours the Termino approach. On the other hand, returning 5 GO documents on the SGD test set and returning 10 on the IC set produces relatively good results. Returning too many documents negatively affects precision more than it positively affects recall, yielding an overall decrease in F measure. Finally, we note that the hierarchical variants of GO documents perform slightly worse than their "flat" counterparts, this may be because general terms introduced with the inclusion of the parent nodes produce too much noise.

5.3 Machine Learning Results

The results for the machine learning approach are shown in Table 3. Comparing the performance of the different classification systems, it can be seen that the DiscoveryNet Naive Bayes implementation performed poorly in terms of recall. This had already been predicted due to the fact that the classifier can only assign one GO term per abstract whereas the two gold standards have on average 2.1 and 6.6 terms per abstract respectively. For both data sets the decision tree classification algorithm in Oracle Text performed best.

Comparing the ML approach to the other methods, it can be seen that for the SGD gold standard ML has lower recall but higher precision. The lower recall is perhaps an indication that many of the GO terms assigned to a paper in the SGD gold standard are in fact directly mentioned in the text and so can best be identified using a lookup approach. The higher precision is expected due to the learn-by-example nature of ML; a GO term will not be assigned to a paper, although it may be directly mentioned in the text, if that GO term does not appear in the data set used to train the classifier.

Although the lexical lookup approach still outperforms the ML approach when using the IC gold standard, the ML approach does show, maybe surprisingly, a significant improvement in recall. This is probably an artefact of the classification algorithm using individual words as the document features: if a GO term is directly mentioned in the document text (as is the case in the IC gold standard), the classifier can learn to strongly correlate the word in the text and the GO term to which the word refers.

6 Conclusions and Future Work

In this paper we have reported on the implementation and evaluation of three simple techniques for assigning GO terms to biomedical abstracts: lexical lookup, IR-type text matching, and text classification based on Machine Learning. GO tagging is an interesting task, both in terms of the text mining challenges it provides, as well as the benefits a working GO tagger can offer to biological scientists and model organism database curators.

Creating a complete and valid Gold Standard from existing resources such as SGD – an issue not addressed by others working on the GO tagging task to date – proved to be more difficult than expected. Despite our imperfect Gold Standards, the results we present in this paper do provide some useful insights into the respective strengths and weaknesses of the techniques we used for GO tagging. The lexical lookup approach is simple and fast, but fails when it comes to recognising variants of terms. Casting GO tagging as an IR problem provides a novel perspective, but treating texts as a bag of words may be too coarse: the discriminatory effect of specific GO terms occurring in GO documents is eclipsed by the occurrence of general terms in these documents, either on their own or as part of a larger GO term. For example terms such as *cell* and *protein* occur far more frequently than specific terms. Similarly, the existence of generic terms in the text and the morphological variations of other terms affects the performance of the ML approaches. Currently, these have been applied without the application of any feature selection or evaluation methods.

Future work will unfold along various lines. First, further effort must go into improving each of the three simple approaches to GO tagging. A possible improvement to the lexical lookup method is to extend the lookup dictionary with terms that are tightly correlated with GO terms. This would be a simple solution to the "term variant" problem. The IR approach could benefit from restricting indices to just noun phrases or (multi-word) terms. Furthermore, the GO document representing a GO term could be augmented with the PubMed abstracts marked with that GO term in one or more model organism databases. Similarly, the ML approaches could directly benefit from the application of both statistical methods and/or the use of domain-specific lexicons for selecting which terms should be used in the feature lists used by a classifier. We are also currently experimenting with various hybrid approaches that can leverage the benefits of each individual approach. The GO terms (and additional biomedical terms such as gene names and MeSH terms) extracted by Termino can be used as features to be fed into the machine learning algorithms. It is expected that using Termino to select and supply a set of feature vectors will both speed up and enhance the classification procedure of the machine learning system.

Secondly, the development of a valid Gold Standard and the use of improved evaluation measures are of critical importance. Our current evaluation measure is based on exact match of GO terms. However, exact match is not the most appropriate criterion for a categorisation task where the labels are drawn from a hierarchically organised set, such as GO. Regarding the creation of a valid Gold Standard, it appears that manual annotation or manual editing of existing resources is inevitable.

Less than perfect text mining results do not necessarily preclude useful and effective end-user applications. For example, to allow users to access and navigate document collections annotated by our GO tagger, we have built a graphical user interface. This provides a tree representation of the GO hierarchy to help the user to zoom in on GO terms of interest. Clicking on a GO term in the tree will display the abstracts that have been assigned that term and the GO terms occurring in the text are highlighted. Such applications need to be evaluated through end-user trials.

Acknowledgements This work was supported by the UK EPSRC research grant: T29369.

References

J.M. Cherry, C. Ball, S. Weng, G. Juvik, R. Schmidt, C. Adler, B. Dunn, S. Dwight, L. Riles, R.K. Mortimer, and D. Botstein. 1997. Genetic and physical maps of saccharomyces cerevisiae. *Nature*, 387(6632 Suppl):67–73.

A. Doms and M. Schroeder. 2005. GoPubMed: exploring PubMed with the Gene Ontology. *Nucleic Acids Research*, 33.

Gene Ontology Consortium. 2000. Gene Ontology: tool for the unification of biology. *Nature Genetics*, 25:25–29.

H. Harkema, R. Gaizauskas, M. Hepple, A. Roberts, I. Roberts, N. Davis, and Y. Guo. 2004. A large scale terminology resource for biomedical text processing. In *Proc. NAACL/HLT 2004 Workshop on Linking Biological Literature, Ontologies and Databases: Tools for Users.*

M. Hearst. 1999. Untangling text data mining. In *Proc. of the 37th Annual Meeting of the Association for Computational Linguistics (ACL99).*

L. Hirschman, A. Yeh, C. Blaschke, and A. Valencia. 2005. Overview of BioCreAtIvE: critical assessment of information extraction for biology. *BMC Bioinformatics*, 6S1.

S. Kiritchenko, S. Matwin, and A.F. Famili. 2005. Functional annotation of genes using hierarchical text categorization. In *Proc. Joint ACL/ISMB Workshop on Linking Biological Literature, Ontologies and Databases: Mining Biological Semantics.*

S. Raychaudhuri, J.T. Chang, P.D. Sutphin, and R.B. Altman. 2002. Associating genes with gene ontology codes using a maximum entropy analysis of biomedical literature. *Genome Research*, 12:203–214.

A. Rowe, D. Kalaitzopoulos, M. Osmond, M. Ghanem, and Y. Guo. 2003. The Discovery Net system for high throughput bioinformatics. In *Proc. 11th Int. Conf. on Intelligent Systems in Molecular Biology.*

T.C. Smith and J.G. Cleary. 2003. Automatically linking MEDLINE abstracts to the Gene Ontology. In *Proc. ISMB 2003 BioLINK Text Data Mining SIG.*