GoTag: A Case Study in Using a Shared UK e-Science Infrastructure for the Automatic Annotation of Medline Documents

M. Ghanem, V. Curcin, Y. Guo

N. Davis, R. Gaizauskas, Y.K. Guo, H. Harkema, I. Roberts

J. Ratcliffe

Dept of Computing
Imperial College London
United Kingdom

Dept. of Computer Science
University of Sheffield
United Kingdom

InforSense Ltd
London
United Kingdom

mmg@doc.ic.ac.uk

n.davis@dcs.shef.ac.uk

j.ratcliffe@inforsense.com

Abstract

In this paper we describe our efforts and experience in constructing GoTag, a distributed system for automatically annotating Medline documents with relevant GO (Gene Ontology) terms. The system is built on top of a service-based text mining infrastructure that integrates tools developed within the Discovery Net and myGrid projects. Two baseline approaches to assigning GO terms have been developed. One assigns GO terms based on directly matching GO term names and synonyms in documents; the other uses a trainable document classifier trained over feature vector representations of documents with which GO terms can be associated using the manually curated yeast genome database. We present preliminary results of evaluating these two approaches and discuss proposals for enhancing both baselines, as well as for constructing a hybrid approach.

1. Introduction

We live in a period of explosive growth of scientific knowledge, expressed very visibly in the growth of the scientific literature. Accessing this literature efficiently and effectively is critical for continued scientific progress. To this end the search for novel or improved automated techniques to support information access and even knowledge discovery in large electronic text collections has become a very active area of research. One part of this research, now popularly referred to as text mining, addresses the goal of extracting or mining information (i.e. "content" or "meaning") at a more abstract level than the literal strings of words by which it is expressed in specific documents. Such extracted information can then be used as a surrogate for the original document or may be used as an annotation on the document to support indexing or clustering or linking or summarisation of documents at the conceptual level. This, in turn, can facilitate searching, browsing and knowledge discovery in general.

Within the EPSRC e-Science programme two pilot projects – Discovery Net and myGrid – recognised the importance of text mining by including text mining capabilities within the more general workflow-based knowledge discovery platforms which were their overall objective. Given this shared interest between the two projects, a follow-on collaborative project was established with two high level objectives: (1) to demonstrate that the service-based text mining components from both projects could be integrated to deliver a more fully functioned text mining capability, and (2) to demonstrate the utility of this combined capability through a case study whose aim is to automatically annotate Medline [1] abstracts with terms from the Gene Ontology (GO) [2] in real time, so as to enable retrieval and clustering of abstracts via the ontology – a novel and useful capability for biological researchers exploring the literature.

In this paper we describe the shared infrastructure we have developed, integrating text mining components from the two contributing projects within a single distributed workflow environment. We also describe two approaches to GO term assignment to Medline abstracts that we have developed within this environment as initial baselines against which further developments can be measured. Evaluating these baselines has meant establishing a corpus of GO-term annotated abstracts which we have done using the manually curated yeast genome database. This approach to

evaluation is not without problems, however, and we discuss issues that arise when trying to establish a gold standard for GO term annotation in this manner. Finally, analysis of the baseline approaches and their performance leads to various proposals for improvement which we aim to address in future work.

2. The Shared Infrastructure

2.1 Discovery Net

The Discovery Net system [3,4] is a workflow-based knowledge discovery environment for the analysis of distributed scientific data. Within Discovery Net, analysis components (whether traditional data mining components or text mining components) are treated as remote services or black boxes with known input and output interfaces described using web service protocols. These services can execute either on the user's own machine or make use of high performance computing resources (Figure 1), such as clusters of workstations, through specialised implementations.

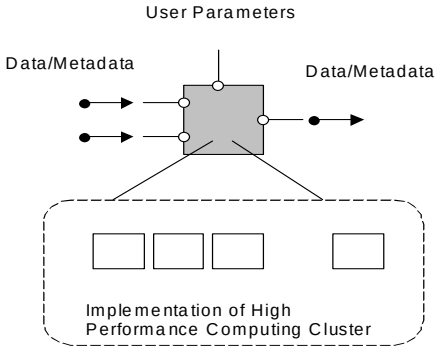

Figure 1 Discovery Net WS interface / HPC implementation of a text mining component

Discovery Net workflows are typically authored through a visual programming interface. Each of the remote web/grid services is first registered within the system, and then the user uses a workflow editor to connect the icons representing these components and the data flow between them. Discovery Net workflows (Figure 2) are expressed in DPML, an XML-based workflow language, and their distributed execution is handled by Discovery Net's workflow execution engine.

In order to support the distributed execution of text mining operations efficiently, Discovery Net supports two data types: an *annotated text* data type, and a *feature vector* type. The annotated text type is based on the Tipster Document Architecture [5] and provides a flexible mechanism for associating properties, or attributes, with text segments of a document that are uniquely defined by their span, i.e. by their starting and

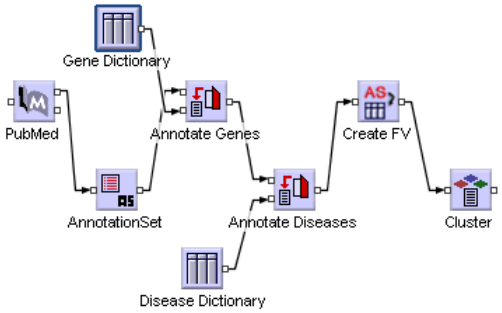

Figure 2 Example Discovery Net workflow based on distributed services

ending position in the document. The use of the annotated text model provides two advantages: first, information computed through earlier stages in a text mining workflow can be passed to later stages without modifying the original text, therefore allowing incremental analysis of the documents. Secondly, since the text itself is not modified by any text mining operation, there is no need to communicate the full text between physically distributed components. This is particularly useful for efficiency purposes when dealing with large document collections that can be mirrored locally to each of the services, in which case the exchange of unique document identifiers does suffice.

The second data type is the feature vector type, in which each document within a collection is reduced to a single numerical feature vector, whose dimensions reflect the significant informational components that the analysis identified. These features can be calculated by a range of services and can be passed to typical data mining operations such as clustering and categorization for further processing. These feature vectors can be represented as either sparse or dense, and the Discovery Net implementation supports both types.

The use of both the annotation model and feature vectors can be explained by referring to the workflow in Figure 2 showing a visual representation of an executable Discovery Net text mining workflow. The workflow performs automatic document clustering using a machine learning algorithm that computes similarity between documents based on the counts of gene names and disease names mentioned in each document. The first component is a data service that retrieves documents from PubMed based on a user supplied query. The second component extracts the texts of the abstracts in the records. The next two components, in turn, use a gene and disease dictionary to identify and mark biological entities appearing in the abstracts and store such information in the annotation structure, which is then passed to a feature vector generation component that counts the occurrences of gene and

disease names in each document. The generated feature vector is passed into a traditional hierarchical clustering component to generate the required document clusters.

2.2 myGrid/AMBIT

The myGrid project has developed a suite of middleware components to support data intensive in silico bio-medical experiments. The core components of myGrid middleware implement a workflow enactment engine which automates sequences of web service processes to perform an experimental function. Amongst the web services that have been made available to the e-biologist are components from a bio-medical text extraction system called AMBIT (Acquiring Medical and Biological Information from Text).

Architecture and Functionality of AMBIT

AMBIT is composed of a number of language processing components, including an information extraction engine, a terminology engine and a free text search engine [6]. The information extraction engine is designed to extract entities, their attributes and relations between them from texts in biomedical domains. It has evolved from the PASTA system [7] and comprises three major stages: lexical and terminological processing, syntactic and semantic processing, and discourse processing.

The first stage includes the recognition and classification of relevant entities that occur in a text, using a terminology engine and a term parser. The terminology engine, called Termino [8], is used to recognise terms using existing terminological resources such as the Unified Medical Language System (UMLS), OMIM, HUGO, etc. The term parser builds longer terms from shorter terms identified by the recogniser according to given term grammars.

The second stage produces a syntactic and semantic analysis for each sentence in the text, linking, e.g. subjects and objects with their governing verbs, which is critical for determining relations between entities.

In the third stage, the discourse module integrates the semantic representations of individual sentences into a semantic representation of the entire text. To do this the discourse module performs co-reference resolution, linking multiple references in the text to the same entity identifier in the semantic model. The discourse module also uses a domain model of background knowledge appropriate to the extraction task, including inference rules that add extra information to the semantic model when certain classes occur in the texts. After the language processing steps are complete the information in the discourse model is read and stored in structured templates, which represent entities, their attributes, and relationships between them.

The information extraction engine in AMBIT is built within GATE, the General Architecture for Text Engineering [9]. GATE implements a model of indexed annotations inspired by the Tipster Document Architecture, also adopted by Discovery Net. Aside from document and annotation management functionality, GATE provides a graphical development environment for modular language processing applications.

Integrating AMBIT in Distributed Workflows

AMBIT is a standalone text mining system whose products are of value only insofar as they can be utilised in further applications so as to assist in the process of information access or knowledge discovery. In order to address this issue we have (1) made a number of AMBIT components accessible via web service interfaces, so that application builders can make use of them (2) constructed several demonstrator applications that suggest how AMBIT services could be used to support information access. The following AMBIT components/products have been made available via web service interfaces:

1. Termino: A client can submit a document together with an indication of which biomedical term classes are to be recognised and the service will return the document with the specified classes marked up.
2. Medline free text search: A client can submit a free text query and the service returns a ranked list of PubMed id's of relevant documents. A secondary service can be used to return the abstracts themselves.
3. Document clustering: A client can submit a list of PubMed id's and the document set can be clustered in various ways, including by MESH category, by GO category (assuming GO codes have been associated with the abstracts) or by chromosome structure (assuming that AMBIT gene-chromosome location relation processing has taken place – see below). In each case the service returns a labelled tree where each node is a cluster of documents.
4. Entity annotated corpus generation: A client can submit a list of PubMed id's and the service returns a corpus consisting of the corresponding abstracts with Termino annotations and an index recording each occurrence of each term across the document set. This is annotated/indexed corpus can be used to support entity-based browsing over a collection of documents (e.g. search results).

A demonstrator application that illustrates the use of these web services as part of a workflow is a

document retrieval and clustering application built as an adjunct to the myGrid workflow demonstrator for Williams Beuren Syndrome (see [13] for details). The fact that AMBIT components have already been made available as web services allows them to be integrated easily into other service-based workflow architectures, such as Discovery Net. This is this route taken in building GoTag.

2.3 The Integrated Infrastructure

We have followed a service-based approach for integrating existing text mining technologies developed within both Discovery Net and myGrid. Existing services are composed and coordinated through the Discovery Net workflow environment workflow model to build new, automated services. The approach is summarised in Figure 4. The workflow paradigm and Discovery Net architecture enables these services to be combined by the workflow author without concern for the origin of the service, Discovery Net and myGrid resources appear to the author as a coherent list of services. The developed workflows can then be deployed as new services, with a configurable user interface, to the end user accessed through a web browser. In our current implementation, services provided by myGrid are invoked remotely from within the Discovery Net workflow using SOAP and a format translation mechanism that allows interoperability between the internal document annotation formats used by both projects.

3. The Case Study and Datasets

3.1 Defining the GO Annotation Task

The task chosen as a case study to validate the integrated text mining infrastructure is to develop a system that annotates Medline abstracts automatically with terms from the Gene Ontology (GO) and in real time to enable retrieval and clustering of abstracts via the ontology. This approach would allow end users to search and browse document collections for concepts rather than terms, allowing them to access information of interest without having to worry about variances in spelling, synonyms or word ordering within phrases.

Using an existing biological ontology to label and organise the concepts within papers has three main advantages: biological scientists should be at least aware of the ontology and may well be skilled with using it; there is no need to go through the process of building a new ontology which can be very time consuming and requires a high level of domain expertise; the organised data can be linked into existing tools for the ontology.

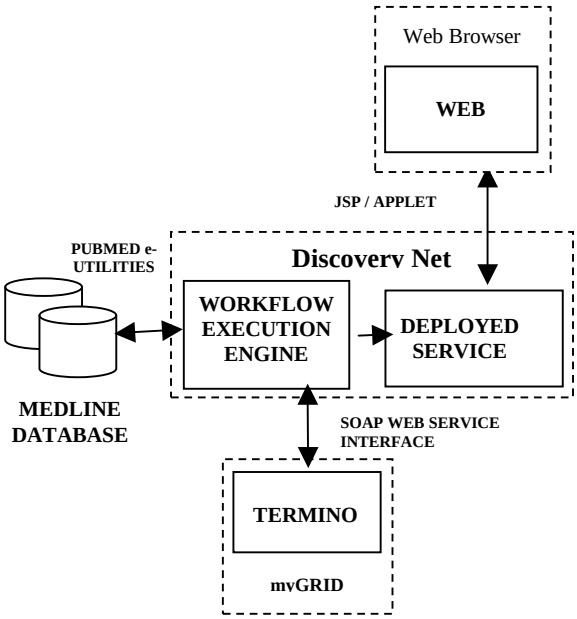

Figure 3 Overview of integrated Discovery Net / myGrid infrastructure

Numerous biological ontologies exist (e.g. the ontologies provided by the Open Biology Ontology consortium http://obo.sourceforge.net/), however the ontology chosen was the Gene Ontology which is a wide-ranging ontology that covers the molecular biology sub-domains of molecular functions, biological processes and cellular components.

3.2 Related Work

Two existing software systems address the problem of assigning GO terms to PubMed documents, GO-KDS [10] which uses a machine learning approach and GoPubMed [11] which uses a term recognition approach.

GO-KDS is a commercial product using a proprietary Weighted Confidence Learner (WCL) classifier to annotate PubMed articles with GO terms at varying levels of confidence. A minimal amount of text pre-processing is performed before classification to remove stop words, expand abbreviations and to parse complex names. Around 26,500 training documents from existing databases that use GO annotations (e.g. SwissProt, Interpro etc.) are used to classify the PubMed documents into approximately 3,700 GO term categories. GO-KDS is available for trial use through a web interface, but no real evaluation of the system is available due to a lack of comparison baseline.

GoPubMed is developed by Dresden University and is based on earlier work from City University in London. The GoPubMed group annotate documents with GO terms by matching the descriptions of GO nodes within papers. However only very rarely does an entire GO node description line appear within a document, so the

semantic weight of each word within the GO node description lines are calculated and the words that carry the most meaning are matched within the document and then mapped back to the original GO terms. Term matching approaches are not without their problems, the main one being that they are reliant on the form of the document being analysed rather than the meaning, so concepts may be missed if they are not articulated using the same terminology as found in GO.

3.3 Available Datasets and Evaluation

The accuracy and effectiveness of a system such as the GoTAg system can be evaluated through the use of manually annotated data sets available from public repositories. Our benchmarks used in this paper are based on the yeast genome database (www.yeastgenome.org) which provides a manually curated mapping between Medline documents and GO terms.

We have collected a set of Medline Abstracts which the SGD database cites as evidence when human experts manually annotate a yeast gene with a GO Annotation. This data set contains 4922 PMIDs and 2455 GO Terms forming 10485 PMID-GO Term pairs. The version of GO we are using contains a total of 18,374 terms arranged as a directed acyclic graph. There is also a stripped down version of GO, called GO-Slim, containing a subset of 87 high-level GO terms.

The evaluation of our GO tagging systems is based on using standard recall and precision measures defined as: *Recall = A/(A+B)*, and Precision = *A/(A+C)*, where *A* is he number of GO-PMID pairs in the evaluation dataset that are predicted, *B* is the number of GO-PMID pairs in the evaluation dataset that are not predicted and *C* is the number of GO-PMID pairs not in the evaluation dataset that are predicted.

The use of the SGD database to provide a gold standard is not without problems. The aim of the SGD curation process is not to tag all the documents in the collection with all the GO terms that appear in these documents, but rather to provide links for each of the genes in SGD to the PubMed papers that were used to infer the GO categories for these genes. Hence the dataset is incomplete and lowered levels of precision are to be expected. Secondly there is no guarantee that a GO term attributed to a paper is mentioned in the paper's abstract. Such terms could have been assigned based on reading the full text of an article and no system (or human) could be expected to assign them to the abstract. This feature of the dataset would affect the maximum recall that could be obtained.

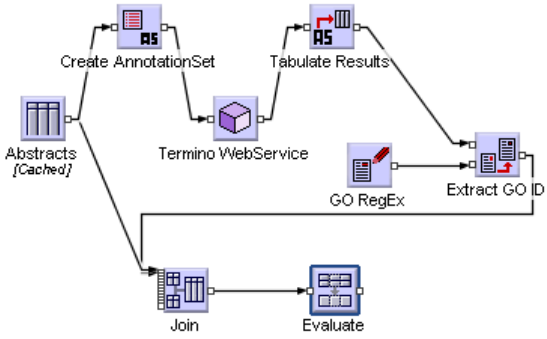

Figure 4 Discovery Net GoTag workflow based on remote calls to TERMINO server

4. Text Mining Methods and Results

4.1 Lexical Look-up Approach

The first approach used in this paper assigns a GO annotation to a document if a GO term, or its synonym, is directly mentioned in the document. This is achieved by performing lexical look-ups invoked through a remote service call to the myGrid Termino engine hosted at Sheffield University. The Discovery Net workflow that implements this approach is shown in Figure 4.

The main nodes in this workflow are implemented using generic components in the Discovery Net text mining repository. They have been labeled by more representative names:

Abstracts Node: a table with four columns storing: *PMID* (unique PubMed identifier), *Title* (document title), *Abstract* (document abstract) and *True GoTAGS* (the set of human assigned GO tags for each document). Each of these columns is stored as text field.

Create AnnotationSet Node: a generic component that tokenizes documents and initializes the *annotated text* data structure for the *Abstracts* columns in the input table, to enable components downstream to add extra information (e.g. the GO annotations) to individual segments within its text.

Termino WebService: a component that makes a remote call to the Termino server, through a web service interface by sending only the text representation of the *Abstract* column for each document in the input table to the remote service. The current implementation iterates through the whole collection by sending one abstract at a time, waiting for a tagged document to be returned from the service, then parsing the returned data structure for the GO tags and adding them to the internal annotated text data structure of Discovery Net.

Tabulate Results Node: a component that creates summary statistics for annotations found in its input. In this workflow, the parameters of the component are set such that the output contains

only two columns, the first containing columns stores a document's *PMID*, and the second column, *Predicted GoTAG*, storing the list of the GO annotations predicted by the remote service.

Extract GO ID, and GO RegEX Nodes: The Extract_GO_ID component performs a simple textual transformation on the *Predicted GoTAG* column to transform the the Termino format (eg: 'go_id:GO:0005574, namespace: cellular_component') to the the more concise representation containing GO term id (eg: 'GO:0005574'). The transformation is based on set of regular expressions provided by the GO_RegEx component.

Join Node: a generic component that performs a "join" operation on two tables to join the *Predicted GoTAG* Column with the original four documents in the Abstracts table, allowing further workflow stages to compare the human assigned tags and the predicted tags.

Evaluate Node: a component that calculates precision, recall and F measures.

Experimental Evaluation of Lexical Look-up

Table 1 summarizes our experiments in using this approach. As could be expected, recall and precision are higher when using the 87 GO Slim terms as opposed to the 18374 GO terms.

GO Version	Recall	Precision	F Measure
GO	15.1%	5.3%	7.8%
GO Slim	51.0%	29.9%	37.7%

Table 1 Precision and recall values for the lexical look-up approach based on Termino

The above precision and recall results may seem poor at the first glance. However, it should be noted that when using the full set of GO terms the large numbers of potential classes to which a document may be assigned make achieving high precision and recall values very difficult. In a traditional classification task, where a document is simply assigned a 'yes' or 'no' class, the likelihood of a correct prediction by chance is 50% (recall and precision would be expected to be 50%). Using GO and GO Slim Annotations as class means the likelihood of a correct prediction by chance are 0.005% and 1.149% respectively (recall and precision would be expected to be 0.005% and 1.149% respectively). The low values can also partly be attributed to the nature of the SGD process that generated the data sets being used, as described in Section 3.3 above.

In order to evaluate the above results better, we have manually analyzed a number of document abstracts and compared the curated vs. predicted tags with the abstract text. Our detailed analysis has revealed the following three main shortcomings of the pure lexical look-up approach:

1. A document may include a direct mention of a GO term that may be represented in the text in a slightly different form from that of the official name and synonyms found in the GO database. For example, The GO term GO:0005739 has the official name 'mitochondrion' and no synonyms. Many of the manually curated documents in the SGD data set that have been assigned this GO term do not include the word 'mitochondrion', but instead include other words such as 'mitochondria'.
2. Although GO terms are organized in a directed-acyclic-graph, they have a hierarchical structure. Terms higher up this structure refer to more general concepts. Because of this, the names and synonyms of the GO terms higher up the hierarchy tend to use some of the same lexical terms used to describe lower concepts. The simple look-up method makes no allowance for this and assigns all GO terms found in paper. For instance a paper containing the text 'mitochondrion distribution' should be assigned GO:0048311 ('mitochondrion distribution'). The lexical look up method will also assign it the GO term GO:0005739 ('mitochondrion').
3. A human expert may read an abstract and infer that it refers to a specific GO term even without explicit mentions of the official GO term, or its synonyms, appearing in the text of the paper. The GO tag can only be inferred from the context of the document of the other words appearing in the document. In this case, there is a semantic relationship between the name of the GO term and the document but no syntactic relationship.

In Section 5 we discuss how some of these shortcomings can be overcome

4.2 Machine Learning Approaches

The second approach to GoTagging that we have explored is based on using machine learning methods. We restricted our experiments to using GO Slim annotations, rather than the full set of GO categories, and used a number of classification algorithms that have been integrated as services in Discovery Net; a Naïve Bayes algorithm, the Rainbow freeware classifier [12], and a Decision Tree and an SVM classifier, both based on Oracle Text. We omit the corresponding workflows for space restrictions.

Evaluation of Naïve Bayesian Classifier

Our first experiment was based on the Discovery Net Naïve Bayes text classifier. The workflow starts by identifying frequently occurring phrases and words in the training data set. A feature vector

is created for each document containing the frequency of these phrases and words in each document. The feature vectors for the training dataset are then passed to the classifier. The results of our experiments are reported in Table 2. The approach offers higher precision, but lower recall, than the lexical look-up approach. The use of single words as features provides the best results. However, one shortcoming of this classifier is that the generated model assigns only one GO annotation to each document, rather than all possible GO annotations. In the evaluation data set each paper has on average 3.5 GO annotations assigned. This limits the maximum recall that can be achieved. The subsequent classifiers overcome this problem.

GO Version	Features	Recall	Precision	F Measure
GO Slim	Words & Phrases	15.8%	54.6%	24.5%
GO Slim	Words	17.8%	61.7%	27.7%
GO Slim	Phrases	16.6%	57.3%	25.7%

Table 2 Evaluation of precision and recall values for the Naïve Bayes classifier

Evaluation of Rainbow Classifier

The Rainbow classifier provides a confidence value for assigning each GO annotation to each document. In our implementation we used only single words as features and filtered out the predicted classes if the confidence of the prediction was lower than a given threshold value. Table 3 summarizes the evaluation of our experiments. As the confidence threshold decreases, it starts to outperform the Naïve Bayes classifier in terms of both precision and recall.

GO Version	Confidence Threshold	Recall	Precision	F Measure
GO Slim	75%	14.0%	59.2%	22.7%
GO Slim	50%	16.4%	58.8%	25.6%
GO Slim	25%	18.9%	57.4%	28.4%
GO Slim	10%	22.5%	57.0%	32.3%
GO Slim	3%	25.8%	55.8%	35.3%

Table 3 Evaluation of precision and recall values for Rainbow Classifier for various confidence thresholds.

Evaluation of Oracle Text Classifier

Our final set of results is based on using Decision Tree and SVM classifiers from Oracle Text that are invoked from within the Discovery Net workflows. Both classifiers return a number of GO annotations per document, rather than a single label. The results are summarized in Table 4, which indicates that the Decision Tree classifiers produces the best precision results across all experiments, but still lower recall than the lexical look-up approach.

GO Version	Classifier	Recall	Precision	F Measure
GO Slim	Decision Tree	36.8%	51.6%	43.0%
GO Slim	SVM	17.5%	53.4%	26.3%

Table 4 Evaluation of precision and recall values for two Oracle Text classifiers

5. Conclusions and Future Work

The main purpose of our joint project has been to investigate the interoperability issues between the e-Science text mining efforts for our two systems. Our infrastructure work has been highly successful in terms of integrating both systems. We hope that these efforts will help us build a reusable text mining infrastructure for the UK e-Science community. We are currently evaluating and investigating the performance optimization issues related to such interoperability.

For the application case study, our initial results show that the lexical look-up approach achieves the highest recall. All machine learning methods achieve the highest precision. In the remainder of this section we briefly describe some ideas for improving the results of each approach, and discuss the possibilities of building hybrid approaches.

5.1 Improving Lexical Look-up Approach

One option to improve upon the shortcomings of the pure lexical look-up implementation, whilst still employing a simple implementation, is to extend the lookup dictionary used. These terms need not be provided by a human expert, but rather can be calculated statistically from the available data sets to find surrogate terms that are tightly correlated with each GO term.

We have conducted various experiments to this effect. Terms tightly correlated with each GO annotation were found by calculating statistics on the word and phrase distribution within the datasets. Chi2 values were calculated for each word and phrase in the document collection against the manually assigned GO Slim classes. These results were then filtered to include only words and phrases with high Chi2 values, meaning they are tightly correlated to specific GO Slim annotations.

Our initial results indicate that this approach can be extremely beneficial. Taking the GO term GO:0005739 (‘mitochondrion’), which was used as an example of the shortcomings of direct GO annotation lookup in Section 4, our statistical methods have found it to be highly correlated with the terms ‘mitochondrial’ and ‘mitochondria’.

5.2 Improving Machine Learning Approach

For the machine learning approaches, which are based on GO Slim, detailed analysis of the misclassification errors reveal that certain default annotations (or classes) consistently contribute to the prediction errors. The reasons are primarily that the training data is skewed towards such annotations. We are currently evaluating various standard machine learning approaches including bagging and boosting and novel centroid based methods to overcome these problems.

We are also currently looking into the use of machine learning approaches for hierarchical classification, which would allow us to classify the whole set of GO terms, rather than concentrating on the set of GO Slim terms.

5.3 Developing Hybrid Approaches

In this paper we mainly investigated using the lexical look-up and machine learning approaches independently. We are currently experimenting with various hybrid approaches that can leverage the benefits of each individual approach.

One particular promising hybrid approach that we have been investigating starts by performing lexical lookups to identify GO Slim terms in the training data. For each GO Slim, correct classification instances are labeled as positive examples and misclassifications are labeled as negative examples. This effectively generates a new training data set for each GO Slim term that is then passed to a machine learning classifier. We are currently investigating this approach based on a different data set based on annotated sentences rather than annotated abstracts

We are also evaluating methods whereby the features vectors to be used in training the classifiers are generated by the Termino server through the identification of MeSH terms and gene names appearing in these documents, rather than simply using features based on words or phrases.

Finally, we are also investigating the use of alternative data sets for training and evaluating our systems. In the current document collection, it should be noted that although the annotations in the dataset are correct, they are not complete (i.e. not all relevant GO terms are manually tagged). This poses a challenge to the models learnt using the automatic machine learning methods,. It also affects our evaluation metrics. We are currently investigating the development and use of a different data collection in collaboration with biologists.

Acknowledgements

We would like to thank our colleagues in both the Discovery Net and myGrid projects for their help and advice. This work has been supported through the “Real-time Text Mining: A Collaboration between Discovery Net and myGrid” grant funded by the EPSRC.

References

[1]. http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=PubMed

[2]. The Gene Ontology Consortium: http://www.geneontology.org

[3]. AlSairafi S et al. The Design of Discovery Net: Towards Open Grid Services for Knowledge Discovery Special issue of *The International Journal on High Performance Computing Applications* on Grid Computing: Infrastructure and Applications, Vol. 17 No 3: 297-315, August 2003.

[4]. Ghanem M et al. A Grid Infrastructure for Mixed Bioinformatics Data and Text Mining. In Proceedings of the *3rd ACS/IEEE International Conference on Computer Systems and Applications*, Cairo, Egypt January 2005.

[5]. TIPSTER www.itl.nist.gov/iaui/894.02/ related_projects/tipster/

[6]. Gaizauskas, R et al, AMBIT: Acquiring Medical and Biological Information from Text. UK e-ScienceAll Hands Meeting 2003.

[7]. Gaizauskas R et al. Protein structures and information extraction from biological texts: the PASTA system. Bioinformatics, Vol. 19, No. 1. (January 2003), pp. 135-143.

[8]. Harkema, H et al. A Large Scale Terminology Resource for Biomedical Text Processing, Proceedings of Linking Biological Literature, Ontologies and Databases, 2004.

[9]. Cunningham, H et al GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications, Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02).

[10]. Smith T. C et al Automatically Linking MEDLINE Abstracts to the Gene Ontology. ISMB 2003 BioLINK Text Data Mining SIG, Brisbane, Australia.

[11]. Doms A et al GoPubMed: Exploring PubMed with the GeneOntology". Nucleic Acid Research Jul 1;33:W783-6.

[12]. McCallum A K. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. 1996. www.cs.cmu.edu/~mccallum/bow

[13]. R. Gaizauskas et al. Integrating Text Mining into Distributed Bioinformatics Workflows: A Web Services Implementation SCC 2004.