Learning Statistical Models for Annotating Proteins with Function Information using Biomedical Text

Soumya Ray[*†]
sray@cs.wisc.edu

Mark Craven[†*]
craven@biostat.wisc.edu

[*]Department of Computer Sciences
University of Wisconsin
Madison, Wisconsin 53706

[†]Department of Biostatistics & Medical Informatics
University of Wisconsin
Madison, Wisconsin 53706

1 Introduction

We participated in the first two subtasks of Task 2 of the BioCreative text mining evaluation. The overall task was designed to evaluate methods for automatically annotating proteins with terms from the Gene Ontology (GO) [Consortium, 2000] using articles from the scientific literature. In the first subtask (2a), a system is given a document, an associated protein and a GO term, and is asked to return a segment of text from the document which supports the annotation of the text with the GO term (the *evidence text*). In the second subtask (2b), a system is given a document and an associated protein, and is asked to return all GO terms that the pair could be annotated with, along with the associated evidence text for each GO term.

Our approach to the annotation task is based on a statistical machine learning perspective. Our approach is fairly straightforward; it incorporates little in the way of linguistic and biological knowledge. It does, however, leverage several existing on-line biological resources, including the MeSH dictionary of biological terms, and databases providing protein-name aliases and GO annotations for proteins. We believe that our approach serves as a useful "baseline" approach, whose performance in the annotation task can likely be improved by the addition of expert biological and linguistic knowledge.

Several key issues need to be addressed to effectively solve Task 2. First, it is unlikely that the exact strings of many GO terms occur in the text that is used to annotate query proteins. It is more likely that the relevance of particular GO terms must be inferred from indirect descriptions that we see in the text. Therefore, we learn models for most GO terms that infer their relevance by looking for related terms. Learning these models, however, calls for more text associated with each GO term than what is available in the Task 2 training set. To address this, we collect data from several publicly available databases that describe GO annotations of protein-document pairs for other (i.e., non-human) organisms. Secondly, even when our GO term models suggest that a GO term might be inferred from a passage of text, we need to evaluate whether this GO term is related the protein of interest. To do this, we learn statistical models to discriminate between passages of text that support GO terms from those that do not. A third key issue is that the relevant passages of text are not marked in the training set. In our approach for BioCreative, we ignored this issue by assuming that all passages that mentioned the protein and a GO term in a training document did in fact relate the protein to the GO term. In current work, we are looking at more sophisticated ways of learning models under the hypothesis that not all of these passages may relate the protein to the GO term.

2 System Description

Processing document-protein queries in our system involves several key steps:

- Documents are pre-processed into a standardized representation.
- The documents are then scanned for occurrences of query proteins. This step involves the use of a protein-alias database and a set of heuristic rules for protein-name matching.
- Selected passages of the documents are then scanned for matches against GO terms. This step employs statistical models of GO terms that are learned from training-set documents.
- Text passages containing putative matches against the query protein and against GO terms are filtered and ranked by a learned statistical model. These models are trained to discriminate between passages that relate GO terms to proteins and passages that do not.

In the following sections, we describe each step in detail. Figure 1 shows a block diagram representation of the overall system.

2.1 Standardizing Documents

The first step performed by our system is to transform a given document into a standardized token-based representation. We first strip all XML tags from the document, while retaining the paragraph structure. We also remove all text outside the abstract and main body of the document. All HTML ampersand codes are transformed to their ASCII equivalents. Next, we remove extraneous whitespace and stem all words using the Porter stemmer

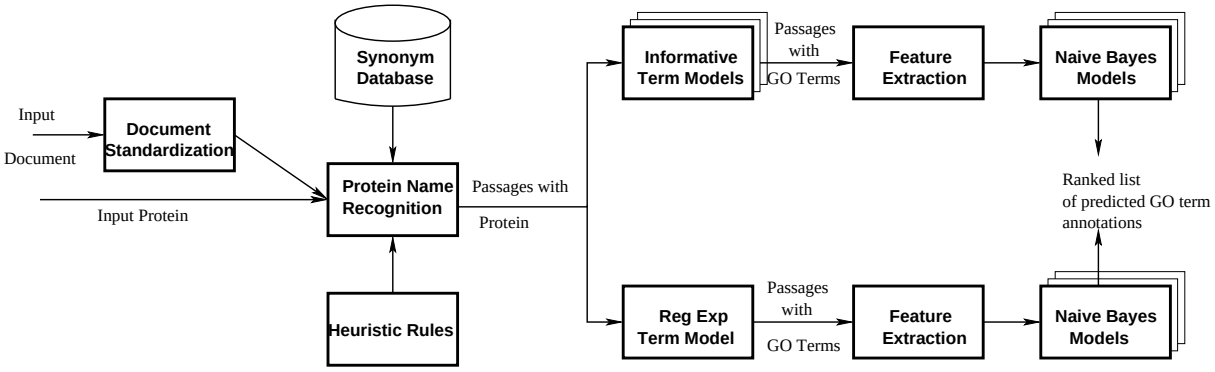

Figure 1: System block diagram for Task 2b. There is one Informative Term Model for every GO term with sufficient training data. There is one Naive Bayes model for each ontology and for each method of GO term prediction (six in all).

[Porter, 1980]. We then transform species names into a common expanded format using a hand-built dictionary of such names. We also split common hyphenated compound words into their constituents using a dictionary of suffixes. Finally, we use a dictionary of biomedical terms from MeSH [National Library of Medicine, 1999] to represent technical compound terms when they occur in a given document. Figure 2 shows an example of the input text before and after the standardizing process.

2.2 Recognizing Protein Names

In order to annotate document-protein pairs with GO terms, we must first find references to a given protein in the document. We do this by searching for the given protein name as well as aliases gathered from SwissPROT [Bairoch and Apweiler, 1997] and HuGO [Wain *et al.*, 2002]. When matching an alias (including the given name) to a piece of text, we use a simple regular-expression representation of the alias as well as the literal string. These regular expressions allow for variations in punctuation and special characters in the matched text.

If we do not find any matches to the given protein name or its aliases, then we search using a set of "approximate aliases" that are generated by applying a set of simple heuristics to the given aliases. Some examples of these heuristics are rules that strip off trailing words (e.g., *protein* and *fragment*), and rules that attempt to reduce a specific protein name to a family name (e.g., by dropping a one-character token at the end of a given name).

2.3 Recognizing GO Terms

In addition to finding references to proteins, our system must also find references to Gene Ontology terms. In many cases, however, we expect that relevant GO terms will not appear verbatim in the articles being processed. Therefore, we construct statistical models to predict whether a GO term is associated with a protein-document pair. In particular, we learn a model for each GO term for which we have sufficient training documents. Since the provided training set is very small and represents relatively few GO terms, we use databases from the GO Consortium website to gather more data. These databases we use include SGD [Dolinski *et al.*, 2003], MGI [Blake *et al.*, 2003], RGD [Steen, 1999] and TAIR [Huala *et al.*, 2001]. They are similar to the GOA database [Camon *et al.*, 2004] given to us in that they list protein, GO term and document code triplets for many proteins belonging to the respective organisms. We extract those triplets from these databases in cases where the associated documents have PUBMED IDs attached to them. Then, we obtain the abstracts for these documents from MEDLINE. We consider this text to be "weakly labeled" with GO terms because it is possible that the evidence associating a GO term of interest to the protein might not be mentioned in the abstract. However, we hypothesize that if we collect significant numbers of documents for any GO term, a large enough fraction will contain this type of evidence, thereby allowing us to learn a model for that GO term. We refer to our models for GO terms as Informative Term models.

Learning an Informative Term model involves identifying terms that are characteristic of a given GO term. To do this we separate our training set into two: a set of articles and abstracts associated with the GO term (the "support" set), and the remaining set of articles and abstracts (the "background"). Then we determine occurrence counts for each unigram, bigram and trigram in our vocabulary in the support text and in the background, and perform a chi-squared test on the table containing

Occurrences of term T in text associated with GO term G	Occurrences of term T in background text
Occurrences of other terms in text associated with GO term G	Occurrences of other terms in background text

Figure 3: Contingency table for the χ^2-test. A high score in the test indicates that it is unlikely that the term T is uncorrelated with the GO term G.

Input: Phospholipases A<INF>2</INF> (PLA<INF>2</INF>)<FNR RID="FN1"> are a rapidly growing family of diverse enzymes that hydrolyze fatty acids at the sn-2 position of phospholipids (<BBR RID="B1">, <BBR RID="B2">).
Output: Phospholipas A2 (Pla2) are a rapidli grow famili of divers enzym that hydrolyz fatti_acid at the sn-2 posit of phospholipid .

Figure 2: Example of document standardization

the counts, as illustrated in Figure 3. This test makes the null hypothesis that the distributions of a term in the two classes (the support and the background) are identical, and returns a score that is proportional to the strength of the alternative hypothesis. Using the returned score, we rank the terms in our vocabulary and pick those whose scores are above an empirically-determined threshold as the Informative Terms for the GO term of interest. As an example, for the GO term GO:0015370, *sodium symporter activity*, this process lists the unigrams *pantothenate, biotin, transporter, lipoate, smvt, uptake,* and *sodium-dependent* as the Informative Terms.

While learning the Informative Term model for a GO term, we are able to take advantage of the hierarchical nature of the Gene Ontology in the following way. We use documents that support a GO term as support for its ancestors in the ontology as well. Thus, documents that were associated with *integral to plasma membrane* are also used when collecting Informative Terms for *plasma membrane*. However, we weight the documents supporting each descendant term proportional to its depth relative to the term under consideration. Thus, documents supporting *integral to plasma membrane* would count for only half as much when calculating evidence for *plasma membrane*. This weight was factored into our calculations during the chi-squared test.

For many GO terms, however, even after collecting "weakly labeled" data, we are unable to accumulate sufficient documents to reliably calculate statistics for the chi-squared test. For such GO terms, we rely on a simple regular expression model. Each regular expression is built from a given GO code name and its aliases.

When given a novel document and protein, we use the Informative Term model to calculate a score for each GO term, based on the Informative Terms that occurred in those paragraphs in the document where the protein name also occurred. A GO term is predicted to be relevant to the document if the score for that term was above a defined threshold, and further, a sufficient number of Informative Terms were matched. For GO terms without Informative Term models, we use the regular expression models described above. A term is predicted to occur if its regular expression matches some piece of text in a paragraph where the protein name also occurs.

2.4 Linking Proteins and GO Terms

Given passages of text that apparently contain references to the query protein and to Gene Ontology terms, we use a statistical model to decide which of these GO terms (if any) should be returned as annotations for the protein.

The Informative Term model does not take into account the actual words in the document beyond the Informative Terms. However, there may be words that are common to many GO terms, that are generally indicative of text supporting the assignment of some GO term to some protein. For instance, words describing localization experiments might be characteristic of text that supports GO terms from the Component Ontology. In order to capture this evidence, we learn two Naive Bayes classifiers for each ontology. Given an attribute-value representation of some piece of text containing a match to the query protein and a GO term, these classifiers return the probability that piece of text supports the annotation of the protein with the GO term. Therefore, they can be used to filter, or re-rank, the outputs of the Informative Term and regular-expression models.

To learn such Naive Bayes classifiers, we first extract features from each paragraph of the document supporting the predictions made by the Informative Term models and the regular expression models on our training set. These features consist of unigrams occurring in the text, as well as several non-specific features which capture the nature of the protein-GO term interaction, such as the length of the passage between the protein and the GO term (or its Informative Terms), the average distance between protein-GO term pairs (if there were more than one) and the score of the match, if the Informative Terms model was used for this prediction. Given a class (supporting/not supporting), a paragraph is described as a multinomial over the vocabulary features and a product of Gaussian distributions over the numerical features:

$$\Pr(D|class) = \prod_{i \in vocab(D)} \alpha_i^{n_i} \prod_{j \in num(D)} \frac{1}{\sqrt{2\pi}\sigma_j} e^{-\left(\frac{x_j - \mu_j}{\sigma_j}\right)^2} \quad (1)$$

where D is the paragraph, α_i represents the probability, given the class, of the i^{th} word from the set of words used in D ($\Pr(w_i|class)$), n_i represents the number of times it occurred, and μ_j and σ_j represent the Gaussian parameters for the j^{th} numeric feature in the given class. We learn a Naive Bayes model for each ontology for GO term predictions made by the Informative Term models, and a separate Naive Bayes model for each ontology for predictions made by the regular expression model.

2.5 Identifying Evidence Text

After the GO term predictions are made by either the Informative Terms model or the regular expression model, the corresponding Naive Bayes model is used to score the

likelihood of each paragraph of the text supporting some protein-GO term association. The maximum score over all paragraphs is then used to re-rank the predictions of the Informative Term and regular expression models. The most highly ranked predictions for that protein and document are then returned by the system.

Our system focuses on predicting a GO term based on the full document given to it, rather than locating a contiguous piece of evidence text for a GO term. Indeed, we feel that it is a strength of our approach that we can aggregate evidence from different regions of a large document in order to make a prediction of a GO term. However, for the purposes of Task 2b and 2a, we were required to identify a single piece of text that provided the best support for a predicted protein-GO term annotation. To do this, we use the following algorithm. We always return a single paragraph as evidence text. If the predicted GO code has an associated Informative Terms model, we use that model to score all the paragraphs in the document where the given protein name occurred. The highest scoring paragraph is then returned. If the GO term is predicted by the regular expression model (or did not have an Informative Term model), we use the Naive Bayes models described in the previous section to rank the paragraphs associated with the predictions, and we return the highest scoring one.

3 Experiments and Discussion

In this section, we present the results of an experimental evaluation of our system based on the initially provided data. For evaluation purposes, we separated the set of documents given to us into a training and a test set. Since we had documents from the *Journal of Biomedical Chemistry* (*JBC*) and *Nature*, these formed a natural partition of the set of documents. Therefore, to evaluate our systems during development, we learned Informative Term models from the known GO term annotations on the *JBC* documents. Then, we used these models and the regular expression models to make predictions of GO terms on the *JBC* documents. We extracted features from these predictions and learned Naive Bayes models using these extracted features. To test our learned models, we used the Informative Term models and regular expression models to make predictions on the documents from the *Nature* journals. We then use our learned Naive Bayes models to rank and filter predicted GO annotations for given proteins and documents. In more recent work (not submitted for evaluation), we have also used logistic regression models instead of the Naive Bayes models to rank and filter predictions.

To evaluate the predictive accuracy of our models, we measure the precision, recall and false positive rates of our predictions. Precision is the fraction of predicted GO term annotations that are correct. Recall is defined as the fraction of correct GO-term annotations that are predicted by the system. The false positive rate is defined as the fraction of false positive predictions in the set of all negatives. The false positive rate is only relevant for the Naive Bayes models, since for these models, the universe of negatives is defined by the predictions of the Informative Term or Regular Expression models.

Experiment	Precision(%)	Recall(%)
Regular Expression	2.1	21.0
Informative Terms	6.1	31.7
Combined	2.9	42.3

Table 1: Precision and Recall results for the system without the Naive Bayes models

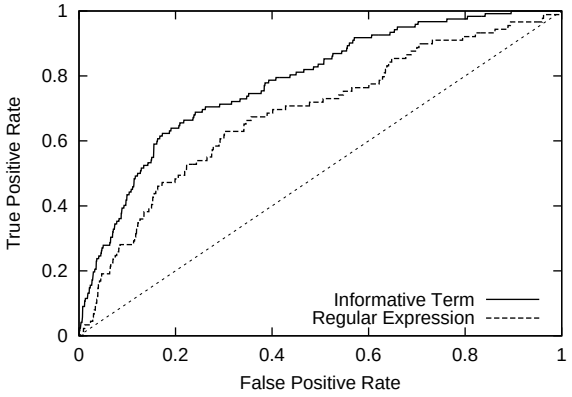

Figure 4: ROC graphs for GO term predictions made by the system aggregated over the three ontologies. The solid line represents predictions made by the Informative Term/Naive Bayes path in Figure 1. The dashed line represents predictions made by the Regular Expression/Naive Bayes path in Figure 1.

First, we evaluated the performance of the system without the Naive Bayes models. For this experiment, we used the Informative Term models and the regular expression models to predict GO terms from the *Nature* articles, and measured precision and recall. These results are shown in Table 1. We observe from the results that without the Naive Bayes models, the system is biased towards recall at the expense of precision. However, we expect to improve our precision by re-ranking these initial predictions using the Naive Bayes models and thresholding the associated confidence. Note, however, that the recall shown for the Combined system in Table 1 is the maximum recall achievable by the system.

To measure the value of the Naive Bayes models, we construct precision-recall (PR) and receiver operating characteristic (ROC) graphs. ROC graphs measure the change of the true-positive rate (recall) of the model against its false-positive rate as a threshold is moved across a measure of confidence in the model's predictions. PR graphs measure the change in precision at different levels of recall. We measure the confidence of a GO term annotation for a protein on a document as the maximum of the posterior probabilities defined by Equation 1 over the paragraphs of the document. In Figures 4 and 5, we show PR and ROC graphs for these

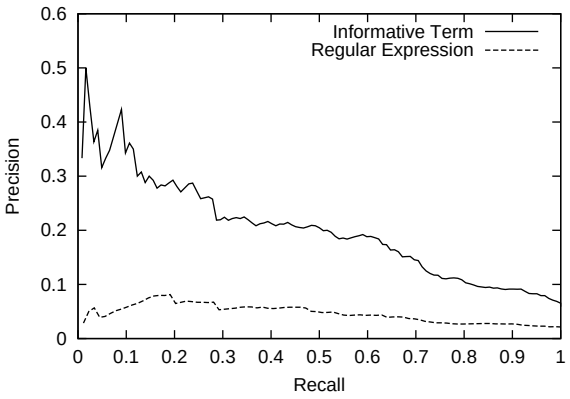

Figure 5: Precision-Recall graphs for GO term predictions made by the system aggregated over the three ontologies. The solid line represents predictions made by the Informative Term/Naive Bayes path in Figure 1. The dashed line represents predictions made by the Regular Expression/Naive Bayes path in Figure 1.

models for predictions aggregated over the Component, Function and Process ontologies, for the cases when the initial predictions were made by the Informative Term models and when the initial predictions were made by the regular expression models. In these figures, the recall (true positive) values are normalized such that values of 1.0 corresponds to the maximum recall values shown in Table 1

From Figures 4 and 5, we observe that the Naive Bayes models are quite effective at discriminating between passages of text that relate proteins to GO terms from those that do not, especially when the initial predictions are made by the Informative Term models. We also observe that when the initial predictions are made by the regular expression models, the Naive Bayes models do not achieve high precision, even at high confidence thresholds. This indicates that (i) there may not be much regularity to be captured in passages supporting these predictions, and/or (ii) our training assumption, where we assumed that each paragraph in the supporting text mentioning the protein and the GO term was actually relating the protein to the GO term, was severely violated, thereby making the data set too noisy for the Naive Bayes learning algorithm to work effectively. We should, however, note that the difference between the Informative Term and Regular Expression models boils down to the availability of training data.

4 Conclusion

We have built a system that uses learned statistical models to automatically annotate proteins with terms from the Gene Ontology based on articles from the scientific literature. Our experimental evaluation of the system indicates that it has predictive value, although there is still much room for improvement. In future work, we plan to investigate several key issues including (i) learning edit-distance based models for recognizing additional instances of protein names, (ii) using models with linguistically richer representations for the step of filtering and ranking candidate annotations, and (iii) using a *multiple-instance* based approach [Dietterich *et al.*, 1997] when learning models for filtering and ranking. The application of a multiple-instance approach is motivated by the fact that, even in the training data, the passages of text that support a given annotation are not marked.

References

[Bairoch and Apweiler, 1997] A. Bairoch and R. Apweiler. The SWISS-PROT protein sequence data bank and its supplment TrEMBL. *Nucleic Acids Research*, 25:31–36, 1997.

[Blake *et al.*, 2003] J.A. Blake, J. E. Richardson, C. J. Bult, J. A. Kadin, J. T. Eppig, and the members of the Mouse Genome Database Group. MGD: The Mouse Genome Database. *Nucleic Acids Research*, 31:193–195, 2003.

[Camon *et al.*, 2004] E. Camon, M. Magrane, D. Barrell, V. Lee, E. Dimmer, J. Maslen, D. Binns, N. Harte, R. Lopez, and R. Apweiler. The Gene Ontology Annotation (GOA) Database: sharing knowledge in Uniprot with Gene Ontology. *Nucleic Acids Research*, 32(1):D262–D266, January 2004.

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

[Dietterich *et al.*, 1997] Thomas G. Dietterich, Richard H. Lathrop, and Tomas Lozano-Perez. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1-2):31–71, 1997.

[Dolinski *et al.*, 2003] K. Dolinski, R. Balakrishnan, K. R. Christie, M. C. Costanzo, S. S. Dwight, S. R. Engel, D. G. Fisk, J. E. Hirschman, E. L. Hong, L. Issel-Tarver, A. Sethuraman, C. L. Theesfeld, G. Binkley, C. Lane, M. Schroeder, S. Dong, S. Weng, R. Andrada, D. Botstein, and J. M. Cherry. Saccharomyces Genome Database. website, 2003. ftp://ftp.yeastgenome.org/yeast/.

[Huala *et al.*, 2001] E. Huala, A. Dickerman, M. Garcia-Hernandez, D. Weems, L. Reiser, F. LaFond, D. Hanley, D. Kiphart, J. Zhuang, W. Huang, L. Mueller, D. Bhattacharyya, D. Bhaya, B. Sobral, B. Beavis, C. Somerville, and SY Rhee. The Arabidopsis Information Resource (TAIR): A comprehensive database and web-based information retrieval, analysis, and visualization system for a model plant. *Nucleic Acids Research*, 29(1):102–105, January 2001.

[National Library of Medicine, 1999] National Library of Medicine. Unified medical language system, 1999. http://www.nlm.nih.gov/research/umls/umlsmain.html.

[Porter, 1980] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):127–130, 1980.

[Steen, 1999] R. G. Steen. RH framework map, Rat Genome Database. website, 1999. ftp://rgd.mcw.edu/pub/maps/rhframework/v.2/.

[Wain *et al.*, 2002] Hester M. Wain, Elspeth A. Bruford, Ruth C. Lovering, Michael J. Lush, Mathew W. Wright, and Sue Povey. Guidelines for human gene nomenclature. *Genomics*, 79(4):464–470, April 2002.