To appear in *Proceedings of the 2004 Text Retrieval Conference (TREC)*. Gaithersburg, MD USA. 2005.

Exploiting Zone Information, Syntactic Features, and Informative Terms in Gene Ontology Annotation from Biomedical Documents

Burr Settles[*†] BSETTLES@CS.WISC.EDU
Mark Craven[†*] CRAVEN@BIOSTAT.WISC.EDU

[*]Department of Computer Sciences, University of Wisconsin, Madison, WI 53706 USA
[†]Department of Biostatistics and Medical Informatics, University of Wisconsin, Madison, WI 53706 USA

Abstract

We describe a system developed for the Annotation Hierarchy subtask of the Text Retrieval Conference (TREC) 2004 Genomics Track. The goal of this track is to automatically predict Gene Ontology (GO) domain annotations given full-text biomedical journal articles and associated genes.

Our system uses a two-tier statistical machine learning system that makes predictions first on "zone"-level text (i.e. abstract, introduction, etc.) and then combines evidence to make final document-level predictions. We also describe the effects of more advanced syntactic features (equivalence classes of syntactic patterns) and "informative terms" (phrases semantically linked to specific GO codes). These features are induced automatically from the training data and external sources, such as "weakly labeled" MEDLINE abstracts.

Our baseline system (using only traditional word features), significantly exceeds median F_1 of all task submissions on unseen test data. We show further improvement by including syntactic features and informative term features. Our complete system exceeds median performance for all three evaluation metrics (precision, recall, and F_1), and achieves the second highest reported F_1 of all systems in the track.

1. Introduction

A major activity of most model organism database projects is to assign codes from the Gene Ontology (The Gene Ontology Consortium, 2000) to annotate the known function of genes and proteins. The Gene Ontology (GO) consists of three structured, controlled domains ("hierarchies," or "subontologies") that describe functions of genes and their products. These domains are Biological Process (BP), Cellular Component (CC), and Molecular Function (MF). Each assigned GO code can also be delegated a level of evidence indicating specific experimental support for its assignment. The TREC 2004 Genomics Track Annotation Task consisted of two subtasks: Annotation Hierarchy, and Annotation Hierarchy Plus Evidence.

We focused on developing a system to solve the Annotation Hierarchy subtask. In this task, we are given a document-gene tuple $\langle d, g \rangle$ and are to determine which, if any, of the three GO domains (BP, CC, MF) are applicable to gene g according to the text in document d. Note that the goal is not to link the gene with a precise GO code (e.g. "Ahr" and "signal transduction"), but rather the domain from which these GO codes come (e.g. "Ahr" and "Biological Process"). This is a significantly simplified version of the annotation task that organism database curators must face, since the exact GO code need not be identified, and we are given the relevant genes a priori.

The training data for this task comes from 178 full-text journal articles as curated in 2002 by the Mouse Genome Informatics (MGI) database (Blake et al., 2003). These documents and their associated genes serve as "positive" examples (i.e. have at least one GO annotation). We are also provided with 326 full-text articles that were not selected for GO curation, but for which genes are associated. These document-gene tuples serve as "negative" examples (i.e. have no GO annotation). In total, there are 504 documents and 1,418 document-gene tuples in the training data. Test data containing 378 documents and 877 document-gene tuples are similarly prepared from articles processed by MGI in 2003.

This paper is organized as follows. First, we provide a fairly detailed description of our system's architecture and explain how more advanced feature sets are generated. We then present experimental results on cross-validated training data, as well as a report on official task evaluation. Finally, we present our conclusions and future directions for this work.

2. System Description

This section describes the architecture of our system. There are several key steps involved in taking a document-gene tuple and predicting GO domain annotations. Figure 1 illustrates the overall system. The following sections describe each step in detail.

2.1 Zoning and Text Preprocessing

First we process the documents, provided by task organizers in SGML format, into six distinct information-content zones: `title`, `abstract`, `introduction`, `methods`, `results`, and `discussion`. Partitioning is done using keyword heuristics during SGML parsing (e.g. "discussion," "conclusion," and "summary" are keywords for the `discussion` zone). We then segment the text in each zone into paragraphs and sentences using SGML tags and regular expressions. All other SGML tags are then stripped away.

Once a document has been zoned and segmented, we apply a gene name recognition algorithm to each sentence. This step involves taking the gene symbol that is provided in the tuple, looking it up in a list of known aliases provided by MGI,[1] and matching these against the text using regular expressions that are somewhat robust to orthographic variants (e.g. allowing "Aox-1" to match against "Aox I" or "AOX1"). We find that this simple matching approach is able to retrieve 97% of all document-gene tuples among positive training documents, and only 52% of tuples among negative training documents. Therefore, we treat this step as a filter on our predictions: documents for which the associated gene cannot be found are *not* considered for GO annotation. Assuming the matching algorithm behaves comparably on test data, this filter sets an upper bound of 97% on recall, but effectively prunes away 48% of potential false positives.

2.2 Feature Vector Encoding

Once the text preprocessing is complete, our system selects only those sentences where gene name mentions were identified. The reasoning for this is two-fold. First, it reduces the feature space for the machine learning algorithms, as we only consider the text that is close to the gene of interest. Second, and perhaps more importantly, it introduces a unique $\langle d, g \rangle$ representation for a document that may contain multiple genes. Consider a single article that is disjointly curated for Molecular Function (MF) of "Maged1," but Biological Process (BP) of "Ndn." By using only sentences that reference each gene, we generate unique feature vectors and can better distinguish between them.

From these sentences, we generate six feature vectors per document (one for each zone). Our baseline system employs traditional bag-of-words text classification features. In this case, each word that occurs in a sentence with a mention of the gene is considered a unique feature (case-insensitive, with stop-words removed). Consider the following sentence, taken from a caption in the ⟨11694502, Des⟩ training tuple:

> "An overlay of desmin and syncoilin immunoreactivity confirmed co-localization."

Our baseline system represents this using features like `word=overlay`, `word=immunoreactivity`, etc. But perhaps the syntactic pattern "OVERLAY OF X" refers to a procedure with some special significance to labeling? Perhaps certain keywords, such as "syncoilin," have semantic correlation with a particular process, component, or function? We wish to induce features that capture such information.

In this paper, we also introduce and investigate two advanced types of features that were used to encode the problem. The features described here are syntactic features (equivalence classes of shallow syntactic patterns) and "informative terms" (n-grams with semantic association with specific GO codes). These features were automatically induced from the training corpus plus several external text sources. Section 3.2 describes the experimental impact of including these features. The following subsections describe how they are generated and incorporated into the system.

2.2.1 External Data Sources

The task training set of 178 MGI-labeled documents provided for the task is decidedly small. To remedy this, we supplement data used for feature induction with training data released for Task 2 of the 2003 BioCreative text mining challenge.[2] This provides an additional 803 documents labeled with GO-related information as curated by GOA (Camon et al., 2004).

While this corpus may be larger, it still represents relatively few specific GO codes (which we use for informative term features). Thus, we also use databases from the GO Consortium website to gather more data about other organisms. The databases we use include FlyBase (The FlyBase Consortium, 2003), WormBase (Harris et al., 2004), SGD (Dolinski et al., 2003), and TAIR (Huala et al., 2001). They are similar to the

[1] ftp://ftp.informatics.jax.org/pub/reports/

[2] http://www.mitre.org/public/biocreative/

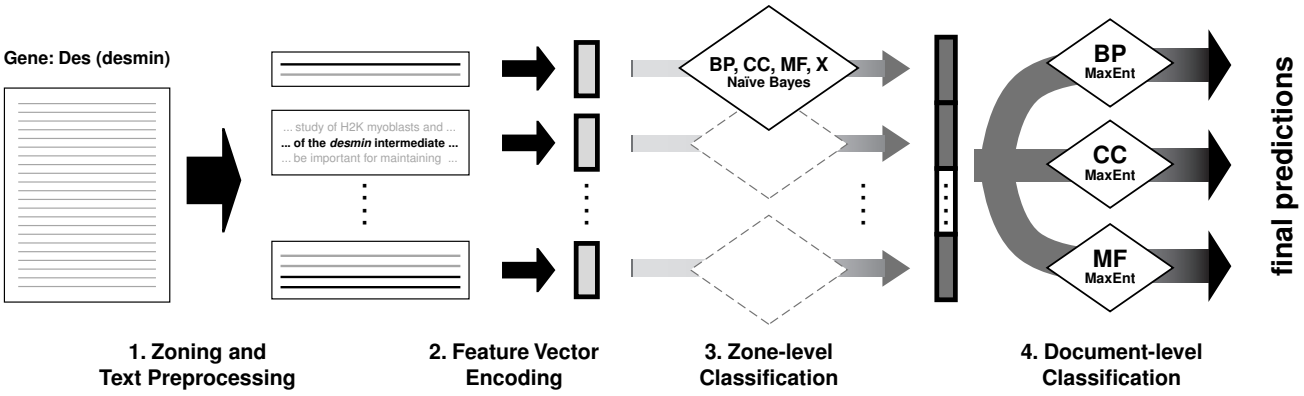

Figure 1. A system diagram for the annotation hierarchy task. (1) Documents are partitioned into zones, and sentences with gene name mentions are identified within each zone. (2) Feature vectors for each zone are constructed using the sentences where gene mentions were found. (3) Label predictions for each zone are made by a trained multi-class Naïve Bayes model, and predictions are combined to create a secondary document-level feature vector. (4) Final binary predictions for each GO domain annotation are made by trained Maximum Entropy models.

MGI curations provided with training data in that they list document, gene, and GO-related triplets for many genes belonging to these respective organisms. We extract the triplets from these databases in cases where the documents have PUBMED IDs associated with them. Then we obtain the abstracts for these documents via a MEDLINE query. We consider these abstracts to be "weakly labeled" with GO codes because they are not complete articles, and thus it is possible that the evidence associating a GO code of interest to the gene might not be mentioned in the abstract. However, we hypothesize that if we collect significant numbers of documents for any GO code, a large enough fraction will contain this type of evidence, thereby allowing us to induce features for that GO code.

PUBMED IDs for all BioCreative data and MEDLINE abstracts are cross-referenced with the TREC test data to ensure that no feature induction occurs on evaluation text. (Even though our models are not directly "trained" on these external sources, the feature induction is still considered part of the training process.)

2.2.2 Syntactic Features

We use syntactic features in an attempt to model deeper structural information about a sentence, which may be useful beyond surface-level lexical features. What we call a syntactic feature is a set of patterns that are information-theoretically clustered to be indicative of the presence of absence of a GO domain.

To generate these patterns, we use a system called AutoSlog-TS (Riloff, 1996), which first constructs a shallow parse of each sentence, then enumerates simple syntactic patterns for subjects and direct objects (e.g. "X BINDS TO", or "TRANSLATION OF X"). The system is run on the training corpus plus external corpora to generate several thousand such patterns. We consider patterns that occur fewer than 10 times not to be useful, and discard them.

We found that treating each pattern as a separate feature has little predictive impact. Thus, we cluster patterns together into a set of disjoint features based on how "purely" they indicate a GO domain. To do this, we separate documents (training data plus external sources) into two sets for each GO domain: the "support" set (documents which are labeled with this GO domain), and a "background" set (all other documents). We then tally the occurrences of each pattern in the support and background texts. For a given pattern p, let $p_{\oplus}$ be proportion of occurrences of p in the support set, and $p_{\ominus}$ be the complementary proportion in the background set. We then rank p using entropy:

$$Entropy(p) = -p_{\oplus}\log(p_{\oplus}) - p_{\ominus}\log(p_{\ominus}).$$

Patterns are then clustered together into features based on (1) whether they occur in greater proportion in the support/background set, and (2) their entropy ranking. We clustered entropy in increments of 0.1 (chosen empirically). Table 1 provides some sample patterns and their statistics for two such features. The pattern "OVERLAY OF X" is clustered into the `cluster=CC_0.0` feature, as it is indicative of the CC label with entropy 0.0 (i.e. totally pure). This feature is assumed to contain syntactic structures that suggest cell components or experimental methodologies relating to them. Likewise, "MOVEMENT FROM

CC @ 0.0 Patterns	CC/Total	Entropy
X INDUCED CURRENTS	24/24	0.000
X INDUCED VESICLES	17/17	0.000
X OVERLAY ASSAYS	11/11	0.000
OVERLAY OF X	11/11	0.000
N-GLYCOSYLATION OF X	10/10	0.000
BP @ 0.2 Patterns	**BP/Total**	**Entropy**
STARVED X	54/56	0.222
ROLLING ON X	25/26	0.235
DEGREES C AT X	24/25	0.242
PATHWAYS TO X	22/23	0.256
MOVEMENT FROM X	19/20	0.266

Table 1. Sample patterns from two syntactic features. Patterns from the feature at the top indicate CC perfectly (entropy = 0), whereas the patterns below are less strongly associated with the BP label.

X" is clustered into the `cluster=BP_0.2` feature, as it indicates the BP label somewhat less strongly, with entropy 0.266. If any of the patterns in a particular syntactic feature occurs in the the text for a zone, that feature is active in the feature vector. This method generated 41 novel syntactic features.

2.2.3 Informative Term Features

Even though the task description is to label documents only with GO *domains*, the specific GO *codes* are actually provided with the training data (and the external sources). Our "informative term" features indicate whether or not there is evidence in the text for a particular GO code lower down in the ontology. We hypothesize that (1) this code-level evidence may help inform a domain-level decision, and (2) if a domain-labeling system can be successfully trained, these features can in turn be useful in making lower level labelings.

To induce informative terms for each GO code, we again separate documents into support and background sets for each GO code c. Then we tally occurrence counts for each term t, which is an n-gram of text (where $1 \leq n \leq 3$), in both of these sets. These counts can be used to construct the following χ^2 contingency table:

# occurrences of term t in code c's support text	# occurrences of term t in code c's background text
# occurrences of other terms in code c's support text	# occurrences of other terms in code c's background text

Any term t with a χ^2 value greater than 200 (empirically chosen threshold) is added to the list of informative terms for GO code c. Table 2 provides some sample informative terms generated for the code GO:0005856 (cytoskeleton, in the CC domain). Similar to syntactic features, if any of these terms occurs in zone text selected for feature vector encoding, the `infterm=cytoskeleton` feature is active for that feature vector. This method generated novel informative term features for 2,739 such GO codes.

Informative Term	χ^2 value
syncoilin	11843.04
filaments	1292.96
microtubule	1278.40
cytoskeleton	720.75
cytoskeletal	715.64
microsphere separation	686.48
dead box proteins	455.41
centrosome	451.85
spindle	396.62
actin binding protein	370.91
latently	344.93
kinesin light chains	332.90
myoblasts and myotubes	296.27
severing	236.60
punctate cytoplasmic pattern	221.93

Table 2. Sample informative terms and their associated χ^2 values generated for the code GO:0005856 (cytoskeleton). A high score indicates that the term is likely correlated with documents discussing the cytoskeleton.

2.3 Zone-level Classification

Feature vectors for each zone are now labeled by a multi-class, multinomial Naïve Bayes model. Naïve Bayes is a well-established machine learning algorithm that has been shown to work well on high-dimensional text classification problems such as this. The model uses Bayes' rule to calculate the probability of label l given evidence in document d. Note that we use the term "document d" here to describe a zone within a document, since classification at this stage is done at the zone level:

$$P(l|d) = \frac{P(l)P(d|l)}{P(d)}.$$

In this framework, $P(l)$ is simply the proportion of training documents that were assigned label l. Since $P(d)$ is held constant for our purposes (it is the same for all labels since the document doesn't change), we must only worry about computing $P(d|l)$.

We use the multinomial Bayesian event model in the fashion of Lewis & Gale (1994). Here we view a document as a series of features, drawn from some feature set. Let C_i be the raw count of the number of times feature f_i occurs in document d. The multinomial event probability for our Naïve Bayes classifier is then:

$$P(d|l) = P(|d|)|d|! \prod_i \frac{P(f_i|l)^{C_i}}{C_i!}.$$

This is a multi-class model, trained to provide probabilities for four labels: BP, CC, MF, and X (no domain). Note, however, that the three GO domain labels are not mutually exclusive. To deal with this, a training tuple which is labeled with more than one domain is repeated during training. Since we are using a Naïve Bayes model, raw counts for features are simply incremented multiple times, once under each label.

The output probabilities of this model are then combined to construct a secondary feature vector. This document-level vector uses 24 real-valued features (4 classes $\times$ 6 zones) which have corresponding semantics (e.g. "the probability of labeling the `title` with BP").

2.4 Document-level Classification

At this stage, we have posterior probabilities for each of four labels assigned to the six different zones in a document. This, in and of itself, is sufficient information to make GO domain predictions. However, we consider the possibilities that (1) evidence for one label may influence the likelihood of annotation for another, and (2) some zones may be more informative for document-level annotation than others.

Thus, the document-level feature vectors we prepare are finally labeled by three binary classifiers, each trained to annotate a different GO domain. In this case, a "positive" example is a document-level vector describing a tuple annotated with the corresponding GO domain in the training data, and a "negative" example describes a tuple annotated with either some other GO domain, or nothing at all.

These secondary document-level classifiers are Maximum Entropy models, for which the probability of a label l for document-level vector d is defined as:

$$P(l|d) = \frac{1}{Z_d} \exp(\sum_i \lambda_i f_i(d, l)),$$

where Z_d is a normalizing factor over all possible labelings of d (to ensure a proper probability in the range [0,1]), and each λ_i is a real-valued weight assigned to feature f_i.

Maximum Entropy is attractive here because it is a discriminative algorithm than does not make the conditional independence assumptions that Naïve Bayes does. It also assigns positive or negative weights to real-valued inputs (in this case posterior probabilities from the Naïve Bayes model), which can be interpreted as positive or negative evidence.[3] There are several nonlinear optimization algorithms than can be employed to find the globally optimal weight setting in training. Our system uses a quasi-Newton method called L-BFGS.

3. Experimentation and Development

We developed our system by conducting various 4-fold cross-validation experiments on the training data. The main system was implemented mostly in Java using, in part, classes from the MALLET library (McCallum, 2002). This library includes implementations of multinomial Naïve Bayes and Maximum Entropy classification models. Evaluation for this task uses three criteria:

$$Recall = \frac{TP}{TP + FN},$$

$$Precision = \frac{TP}{TP + FP}, \text{and}$$

$$F_1 = \frac{2 \times Recall \times Precision}{Recall + Precision},$$

where TP means true positives, FP means false positives, FN means false negatives, and F_1 is the inverse harmonic mean of precision and recall, a summary statistic to account for the inherent trade-off between them.

3.1 Exploiting Zone Information

First of all, we hypothesized that using zone information would be generally useful in making document-level annotations where the location of relevant textual content is unknown. Our first step was to investigate the validity of this idea. Figure 2 compares recall-precision curves for three systems: (1) a Naïve Bayes model trained on words from the entire document and ranked by posterior probability, (2) a similar model divided into zones (each zone submitting a "vote" for classification weighted by its posterior probability), and (3) the two-tier classification described in Section 2. From this figure we can see that moving

[3] We also experimented with Support Vector Machines for the document-level model, which perform comparably.

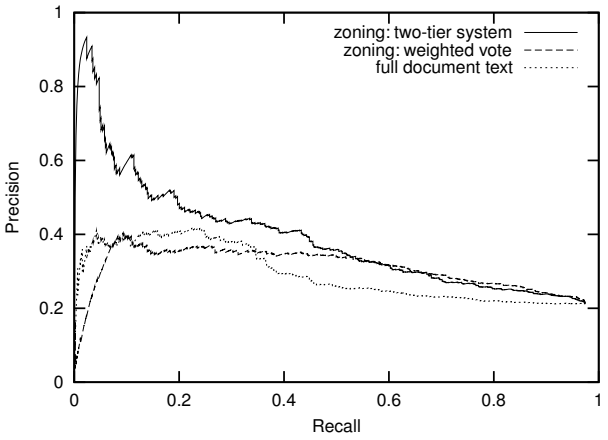

Figure 2. Recall-precision curves comparing use of zoning information in various ways during system development (using 4-fold cross-validation).

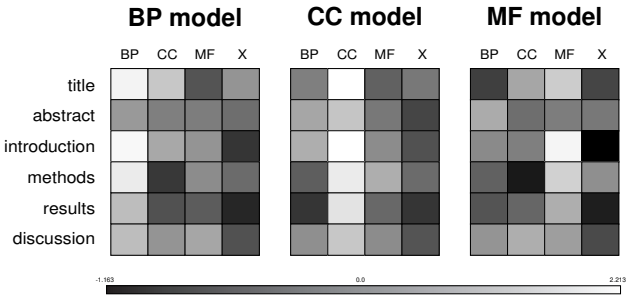

Figure 3. A visualization of weights assigned by each document-level Maximum Entropy model to the outputs of the zone-level Naïve Bayes model. Lighter cells indicate positive weight, darker cells indicate negative weight, and grey cells have weight close to 0.

from a full-document classification model to a simple vote zoning system has a slight impact: sacrificing precision at low levels of recall, but maintaining higher precision at high levels of recall. However, moving on to the two-tier system significantly improves precision across the whole recall spectrum.

Figure 3 is a visualization of the weights assigned by each document-level Maximum Entropy model to the posterior probabilities of each zone-level labeling. We can see that a zone-level CC label is generally a good predictor of a document-level CC label (likewise for BP and MF), and that X (no annotation) weights are generally negative. Moreover, we see stronger weights for the `title` and `introduction` zones, indicating highly predictive information in these text regions, while `abstract` and `discussion` seem surprisingly uninformative.

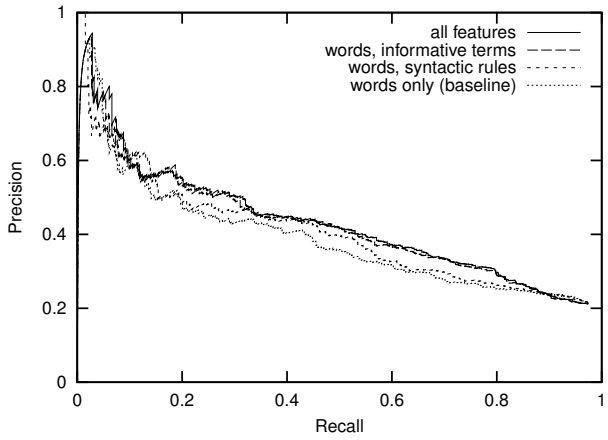

Figure 4. Recall-precision curves comparing use of different feature sets during system development (using 4-fold cross-validation).

3.2 Varying the Feature Set

We also wanted to evaluate the impact of using different subsets of the advanced features described in section 2.2 to extend our baseline system. Figure 4 compares recall-precision curves for the system using various features sets. In general, we see that both syntactic features and informative terms improve precision for a given level of recall, and combined they show even more improvement. This turns out to hold true for official track evaluation data as well.

4. Official Evaluation Results

Using the results obtained during development, we select the predictive threshold for each Maximum Entropy model that optimizes F_1 for that GO domain on the 4-fold cross-validated training data. Models are then re-trained on the entire training corpus and applied to the unseen test corpus for official evaluation.

Table 4 illustrates how the four runs from our system, each using a different subset of features, perform in the official task evaluation. We also report minimum, median, and maximum scores for all track participants. Our top two systems surpass median recall, our top three surpass median precision, and all four systems achieve a substantially higher F_1 than the median for the track. (Our best system comes close to the overall maximum). Figure 5 presents the results of official runs, for all track participants, in recall-precision space.

System description	R	P	F_1
words only (baseline)	55.96	39.35	46.21
words, syntactic features	55.96	42.55	48.34
words, informative terms	62.63	42.18	50.41
all features	62.02	43.86	51.38
track min	13.33	16.92	14.92
track median	60.00	41.74	35.84
track max	100.00	60.14	56.11

Table 3. Comparison of our system using four different feature sets on official TREC evaluation data. Also presented are the minimum, median, and maximum scores for each metric for all task participants.

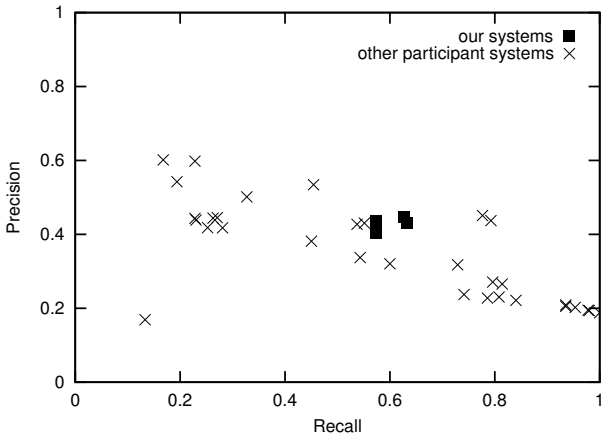

Figure 5. Graphical comparison in recall-precision space of our systems vs. all other system runs submitted for official task evaluation.

5. Conclusions and Future Work

The performance of our two-tier system for the TREC Annotation Hierarchy task seems very promising. The major goals of this study were to investigate the value of (1) zone information, and (2) advanced features not often used in such text classification tasks (specifically syntactic features and GO-specific informative terms). This early work suggests that both facets can offer substantial gains over our baseline approach for this problem.

However, because the present work was greatly constrained by time (all work was done in approximately six weeks), there are many things we were unable to investigate. Among other things, the single Naïve Bayes model ⇒ multiple Maximum Entropy model architecture here is a bit theoretically unfounded. We did experiment with using unique Naïve Bayes models for each zone with suboptimal performance (though this may be due to data sparsity). We also used Maximum Entropy models for zone-level classification, but experienced severe overfitting. In short, we were able to get the present architecture to work empirically within the task time line, but would like to try more variants of this two-tier architecture.

Exploiting zone information—at least as it is presented and used here—is quite helpful. However, our notion of "zone" and the methods by which we partition a document into them are necessarily ad-hoc. It would be interesting to see if incorporating a finer grained, more rhetorical zone ontology (Mizuta & Collier, 2004) is more helpful. We are currently investigating machine learning approaches to automate this kind of zoning, and plan to incorporate this into the Annotation Hierarchy task as well.

The patterns we generate with AutoSlog-TS are simple and based on deterministic parses more suited to general English than the vernacular of biomedical journals. The system also generates all arbitrary patterns, rather than those that extract genes and proteins specifically. We would like to see if refining the parsing, pattern generation and rule clustering can lead to even more useful syntactic features. The informative term features presented here are generated similar to previous work (Ray & Craven, 2005), but they are used quite differently. We consider the occurrence of an informative term to be a multinomial event (similar to word features). However, much richer information is available to us: textual distance from the term to the gene of interest, the χ^2 confidence of that particular term, etc. It is our belief that this sort of information would be useful to the model as well.

Finally, curation and annotation tasks of this nature seem ideal for research in active learning. We intend to investigate algorithms that can learn to make predictions at the zone or passage level (rather than the document level) and garner feedback from human oracles.

Acknowledgements

This work was supported in part by NLM grant 5T15LM007359 and by NIH grant R01 LM07050-01. Special thanks to Soumya Ray for his advice, help in collecting and processing external data sources, and assistance generating the informative term features. We would also like to thank the TREC 2004 Genomics Track organizers for their hard work making the task possible.

References

Blake, J., Richardson, J. E., Bult, C. J., Kadin, J. A., Eppig, J. T., & the members of the Mouse Genome

Database Group (2003). MGD: The Mouse Genome Database. *Nucleic Acids Research*, *31*, 193–195. http://www.informatics.jax.org.

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

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

Harris, T. W., Chen, N., Cunningham, F., Tello-Ruiz, M., Antoshechkin, I., Bastiani, C., Bieri, T., Blasiar, D., Bradnam, K., Chan, J., Chen, C.-K., Chen, W. J., Davis, P., Kenny, E., Kishore, R., Lawson, D., Lee, R., Muller, H.-M., Nakamura, C., Ozersky, P., Petcherski, A., Rogers, A., Sabo, A., Schwarz, E. M., Auken, K. V., Wang, Q., Durbin, R., Spieth, J., Sternberg, P. W., & Stein, L. D. (2004). WormBase: a multi-species resource for nematode biology and genomics. *Nucleic Acids Research*, *32*, D411–D417.

Huala, E., Dickerman, A., Garcia-Hernandez, M., Weems, D., Reiser, L., LaFond, F., Hanley, D., Kiphart, D., Zhuang, J., Huang, W., Mueller, L., Bhattacharyya, D., Bhaya, D., Sobral, B., Beavis, B., Somerville, C., & Rhee, S. (2001). 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*, 102–105.

Lewis, D., & Gale, W. (1994). A sequential algorithm for training text classifiers. *Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 3–12). ACM/Springer.

McCallum, A. (2002). MALLET: A MAchine Learning for LanguagE Toolkit. http://mallet.cs.umass.edu.

Mizuta, Y., & Collier, N. (2004). Zone identification in biology articles as a basis for information extraction. *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA)* (pp. 29–35). Geneva, Swizterland.

Ray, S., & Craven, M. (2005). Learning statistical models for annotating proteins with function information using biomedical text. *BioMed Central (BMC) Bioinformatics*. In Press.

Riloff, E. (1996). Automatically generating extraction patterns from untagged text. *Proceeding of the Third National Conference on Artificial Intelligence* (pp. 1044–1049). MIT Press.

The FlyBase Consortium (2003). The FlyBase database of the Drosophila genome projects and community literature. *Nucleic Acids Research*, *31*, 172–175. http://flybase.org/.

The Gene Ontology Consortium (2000). Gene Ontology: Tool for the unification of biology. *Nature Genetics*, *25*, 25–29.