Predicting Gene Functions from Text Using a Cross-Species Approach

Emilia Stoica and Marti Hearst

Pacific Symposium on Biocomputing 11:88-99(2006)

PREDICTING GENE FUNCTIONS FROM TEXT USING A CROSS-SPECIES APPROACH

EMILIA STOICA AND MARTI HEARST
SIMS, UC Berkeley
estoica@sims.berkeley.edu, hearst@sims.berkeley.edu

We propose a cross-species approach for assigning Gene Ontology terms to LocusLink genes based on evidence extracted from biomedical journal articles. We make use of information from orthologous genes to derive and merge two sets of GO codes for a given target gene. For the first set, we restrict GO code assignments to be selected from only those codes which have already been assigned to the target gene's ortholog. Since this approach results in high precision but low recall, for the second set, we allow any GO code to be a candidate, but then eliminate those codes which are illogical to pair with a GO code that is known to be associated with the orthologous gene. Experimental results on three datasets show that the F-measure obtained with this algorithm is consistently higher than the F-measure of other current solutions.

1. Introduction

The complexity of molecular biology is reflected in the large number of experimental results reported in MEDLINE documents, which provide valuable information about the functions of genes and gene products. Extracting these functions from literature (also known as functional annotation), may be a step forward toward understanding diseases and identifying drug targets [4].

Given the large variability in expression of concepts in medical literature, researchers have created a common language for functional annotation, the Gene Ontology (GO)[9]. GO is a controlled vocabulary of over 17,600 terms, also known as GO codes. Each GO code consists of tokens, which are words or punctuation characters. GO codes are organized into three distinct direct acyclic graphs, corresponding to molecular functions (*MF*), biological processes (*BP*) and cellular components/locations (CC) of gene products. More general terms act as parent nodes of the less general ones. For example, the GO code *development* (GO:0007275) is the parent of *embryonic development* (GO:0009790), which in turn is the parent of *somitogenesis* (GO:0001756).

Extracting gene functions from literature is currently done manually, a laborious and time consuming process. Human curators read each document and

annotate genes with GO codes if the text contains evidence that supports the annotation[2,5]. Given the enormous number of publications in MEDLINE, manual curation cannot keep pace with the data generation. However, automatic functional annotation is a challenging task, for the following reasons, among others:

(1) When a GO code is assigned to a gene, its GO tokens may not explicitly occur in the text. For example, in document (with PubMed Id) 11401564, GO code *3'-5'-exoribonuclease activity* (GO:0000175) occurs as *3' to 5' exoribonuclease activity*, while in document 11110791 occurs as *3' → 5' exoribonuclease activity*. Similarly, in document 10692450, GO code *negative regulation of cell proliferation* (GO:0008285) occurs as *inhibition of cell proliferation*.

(2) GO tokens do not necessarily appear contiguously in the annotated text. For example, in document 10734056, gene *MIP-1 alpha* is annotated with GO code *G-protein coupled receptor protein signaling pathway* (GO:0007186), based on the following paragraph: *Results indicate that CCR1-mediated responses are regulated ... in the signaling pathway, by receptor phosphorylation at the level of receptor/G protein coupling. ...CCR1 receptor binds MIP-1 alpha with high affinity*.

(3) Algorithms that attempt to assign GO codes to documents based just on the fact that the tokens from the GO codes occur in the text, yield a large number of false positives. Even when the GO tokens occur in text, the curator may not annotate the gene with the GO code because (a) the text does not contain evidence to support the annotation, or (b) the text contains evidence for the annotation, but the curator knows the gene to be involved in a function that is more general or more specific than the GO code that was matched in the text. For example, the Gene Ontology provides guidelines of what the evidence for annotation should be, (e.g., the text should mention *co-purification* or *co-immunoprecipitation* experiments). However, an algorithm that uses this information (e.g., annotates a gene with a GO code only if the text contains words like *co-purification*) does not perform any better than an algorithm that ignores these hints about evidence.

To address these challenges, we propose a cross-species approach for assigning Gene Ontology terms to LocusLink genes, making use of information about orthologous genes. (Orthologous genes are genes from different species that have evolved directly from an ancestral gene.) Our assumption is that since there is an overlap between the genomes of the two species, their orthologous genes may share some functions, and consequently, some GO codes.

We use information from orthologous genes in two ways. First, for a target gene we search in biomedical journal text for only the GO codes previously assigned to its orthologous gene. This yields precise results but at the expense of missing many codes. In the second method, for a given gene we search in biomed-

ical text for any of the 17,600 possible GO codes, but eliminate those codes that are illogical, based on which GO codes are known to co-occur with the GO codes for the ortholog of the gene. This approach is less precise but uncovers more valid codes. We then merge the results of the two processes. Results on three datasets show that our algorithm obtains higher F-measure than previous solutions.

The rest of the paper is organized as follows. Section 2 presents related work. Section 3 describes our solution. Section 4 presents experimental results, and Section 5 concludes and suggests future work.

2. Related Work

Functional annotation from medical documents is a relatively new problem, although there is significant related work for annotating a gene with functions using gene expression time profiles[12,16], sequence-derived protein features[17] and multiple alignments of complete sequences[6]. Many approaches search for uncharacterized sequences across GO-mapped protein databases and assign to them the GO codes of the best hits[13,19,27].

Functional annotation from bioscience articles has been mainly studied by the participants in the BioCreAtIve[15] and the TREC Genomics track[14] competitions. BioCreAtIve addressed the problem of annotating a gene with the exact GO codes and thus has created a defacto benchmark for functional annotation from bioscience literature. TREC made the task easier; rather than exact GO codes, participants had to predict the GO category (molecular function, biological process or cellular component) the GO code belongs to. Below we summarize the methods proposed by the participants in the BioCreAtIve competition.

Chiang and Yu[7,8] observe that there are phrase patterns commonly used in sentences describing gene functions. Examples are *"gene plays an important role in function"*, or *"gene is involved in function"*. To learn the patterns they divide a sentence s into five segments (*prefix, tag1, infix, tag2, suffix*), where *tag1, tag2* are gene products or functions. The *prefix, infix* and *suffix* are divided into tokens and the patterns are learned by seeking out consecutive tokens common to multiple sentences. To predict the overall likelihood that a sentence describes a gene-function relation, they use a Naive Bayes classifier.

Ray and Craven[23] learn a statistical model for each GO code from a training set of four GO annotated databases. In particular, they learn which words are likely to co-occur in the paragraphs containing the tokens of a GO code. They use a multinomial Naive Bayes classifier for every GO category to re-rank the results from pattern matching. Features are words, as well as the distance between the protein and the GO code in the text, and the score of the match.

Couto et al.[10] annotate a gene with a GO code if what they call the *information content* of the GO code, computed as a function of the words that match in text, is larger than its information content computed as a function of all the words in the GO code. Verspoor et al.[26] compute an association strength between words based on how often they co-occur in the paragraphs of a set of documents. Every GO code is expanded with the words having a high association strength with the words in the GO code. GO codes are assigned to genes using a Gene Ontology Categorizer[18] which utilizes the structure of the Gene Ontology to find the best covering nodes.

Ehler and Ruch[11] treat each document as if it was a query to be categorized into GO categories. GO codes are assigned scores based on pattern matching and $TF \times IDF$ weighting and the top GO codes are annotated to the gene. Rice et al.[25] learn a support vector machine classifier for each GO code. Target genes are tested against each classifier and are assigned the highest scoring GO codes.

The literature contains a few other solutions for functional annotation, although these systems did not participate in the BioCreAtIve competition. Raychaudhuri et al.[24] compare three document classification techniques (Maximum Entropy Modeling, Naive Bayes and Nearest Neighbor) for assigning only 21 GO codes to gene products. Koike et al.[21] use shallow parsing and rule-based techniques to semi-automatically enrich GO codes with other terms that appear in the same sentence based on co-occurrence and collocation similarities. Finally, Xie et al.[28] combine both text mining and sequence similarity searches to annotate gene products with GO terms. The results are reported on various datasets, thus it is difficult to compare our solution against them.

3. Algorithms

In this section, we describe our algorithms for annotating genes with GO codes. We make use of information from orthologous genes to derive two sets of GO codes for a given target gene. For the first set (called CSM, **C**ross **S**pecies **M**atch), we restrict GO code assignments to be selected from only those codes which have already been assigned to the target gene's ortholog. Since this approach results in high precision but low recall, for the second set (called CSC, **C**ross **S**pecies **C**orrelation), we allow any GO code to be a candidate, but then eliminate assignments that cannot pair with the gene's ortholog. The final set of annotations is the union of the two sets. Figure 1 shows the block diagram of the annotation process.

In every document, we eliminate stop words and punctuation characters and divide the text into tokens using spaces as delimiters. We analyze text at the

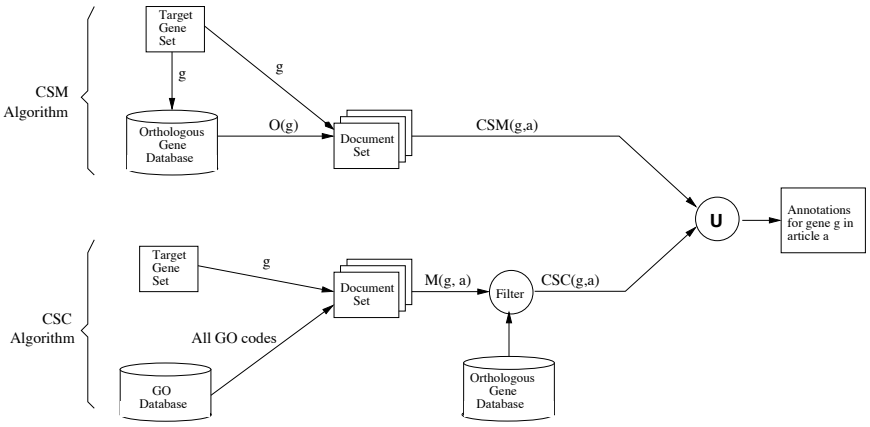

Figure 1. The annotations for gene g computed as the union of two sets CSM and CSC.

sentence level. Similarly, we divide GO codes into tokens. We perform gene name recognition by normalizing and matching different variations of gene names using the algorithm of Bhalotia et al.[1] For every sentence in which a target gene is found, we consider a GO code to be found if the sentence contains a percentage of tokens from the GO code that is larger than a threshold. This threshold is set to 75% for the CSM algorithm, and 100% for the CSC algorithm.

3.1. *CSM: Using the GO codes of Orthologous Genes*

The GO ontology contains 17,600 GO codes (as of July 12, 2004). Our experimental results show that searching in text for all the GO codes results in a large number of false positives and thus low precision. For this reason we aim to limit the set of GO codes that are possible candidates. We achieve this by searching in text for only the GO codes previously annotated to orthologous genes.

As mentioned above, the main assumption behind this algorithm is that for two species that have descended from a common ancestor, the orthologous genes of the two species may have the same functions, and consequently may be annotated with the same GO codes.

For a target gene g, let $O(g)$ represent the set of GO codes that have been assigned to the ortholog of that gene for another species. For a given article a, this algorithm finds all sentences that contain the gene g and then searches only for those GO codes in $O(g)$. We define $CSM(g, a)$ to be the subset of GO codes in $O(g)$ matched in article a for gene g.

It is important to note that many genes are annotated by automated or man-

ual transfer of annotations from other genes with sequences similar to the target genes. Such annotations are marked with the evidence codes *Inferred from electronic annotation (IEA)* and *Inferred from Sequence Similarity (ISS)*. While these annotations are very useful, using them in our case may unrealistically boost our performance. This is because in some cases the annotations of an orthologous gene may have been derived from the annotations of the target gene. To avoid this kind of circular reference, we do not use any annotations of orthologous genes marked with the evidence codes *IEA* and *ISS*.

3.2. *CSC: Using All GO Codes and Eliminating "Illogical" Ones*

Although searching in text for only the GO codes of orthologous genes yields high precision, it limits recall since these codes are only a small subset of those available. To improve recall we use a general observation: if two GO codes tend to occur together in a database, then a gene annotated with one GO code is likely to be annotated with the other one as well. Similarly, if one GO code tends to occur in the orthologous genes' annotations when another does not, then for the target species these two GO codes may not be allowed to both be assigned to the same gene.

The idea is that GO codes co-occur if it makes sense for a gene to support both of their functions; in many cases the underlying biological function will make it illogical for two codes to co-occur. For example, if we find *rRNA transcription* (GO:0009303), *nucleolus* (GO:0005737) and *extracellular* (GO:0005576), then we eliminate *extracellular* because transcription cannot happen outside of the cell.

King et al.[20] use a similar idea to predict how to augment those GO codes that have already been assigned to a gene, once some annotations for the gene are known. Given a database of genes and their GO annotations, they use machine learning algorithms trained on one part of the dataset to predict the annotations for the rest of the database. They do not use cross-species information, nor do they use the correlations to find GO codes in text.

For every pair of GO codes in the orthologous genes database, we compute a χ^2 coefficient[22] using occurrence counts. Let N be the number of GO codes and:

O_{11}: # of times the orthologous gene is annotated with both GO_1 and GO_2
O_{12}: # of times the orthologous gene is annotated with GO_1 but not with GO_2
O_{21}: # of times the orthologous gene is annotated with GO_2 but not with GO_1
O_{22}: # of times the orthologous gene is not annotated with any of GO_1 or GO_2

Then the χ^2 coefficient is

$$\chi^2 = \frac{N * (O_{11} * O_{22} - O_{12} * O_{21}) * (O_{11} * O_{22} - O_{12} * O_{21})}{(O_{11} + O_{12}) * (O_{11} + O_{21}) * (O_{12} + O_{22}) * (O_{21} + O_{22})}$$

$CSC(g,a) = \{\}$;
for every GO_1 in $M(g,a)$
    $count = 0$;
    for every GO_2 in $O(g)$
        if $((\chi^2(GO_1, GO_2) > 3.84)\&\&(GO_1 \neq GO_2))$
            $count++$;
    if $(count > p * o)$
        add GO_1 to $CSC(g,a)$;

Figure 2. *Pseudocode for computing set CSC for gene g in article a.*

For every gene g in article a we search for all 17,600 GO codes. Let $M(g,a)$ be the set of GO codes matched and let o be the size of $O(g)$. Also let $CSC(g,a)$ (**C**ross **S**pecies **C**orrelation) be a set of initially empty annotations for gene g in article a. Figure 2 shows the algorithm for computing the set $CSC(g,a)$.

For every GO code GO_1 in $M(g,a)$, we count how many GO codes GO_2 in $O(g)$ have a χ^2 coefficient larger than 3.84[a]. If the count is larger than o multiplied by a percentage p (0.2 in our experiments[b]) then we consider GO_1 logically related to the GO codes in $O(g)$, and we add it to the set $CSC(g,a)$. Otherwise it is discarded. The final set of annotations for gene g in article a is the union between sets $CSM(g,a)$ and $CSC(g,a)$.

4. Results

In this section we present experimental results. We test our algorithms on the dataset of task 2.2 of BioCreAtIve competition[3], where we compare our results with the performance of the participants in the contest. In addition, we test our algorithms on two other GO annotated databases: EBI human[c] and MGI [d].

4.1. *Results on the BioCreAtIve Dataset*

Task 2.2 of the BioCreAtIve competition provided participants with a set of gene-article pairs and asked them to annotate the genes with the GO codes found in the articles along with the passages supporting the annotations.

[a]For a probability level of 0.005, and one degree of freedom the probability of error threshold for χ^2 is 3.84[22].
[b]Intuitively, we may expect higher percentages to work better. However, since genes may be involved in several unrelated functions, a GO match in text is generally correlated with a small percentage of functions in $O(g)$.
[c]http://www.ebi.ac.uk/Databases.
[d]http://www.informatics.jax.org.

The test set consisted of 138 human genes and 99 full text articles. Human curators judged each annotation. An annotation was marked as "perfect prediction" if the gene name appears in the retrieved passage of text and the passage provides evidence for annotating the gene with the GO code. There was no official evaluation measure but the committee of judges reported, for each system, the total number of predictions, the number of perfect predictions and precision. In total, participants found 237 "perfect predictions". Since the competition organizers did not report numbers for recall, we use the number 237 as the total number of relevant documents for computations of recall.

We conducted this research after the contest had past, so our annotations could not be judged by human curators, which makes it impossible to fully determine how well our performance compares with the other systems. To get around this limitation, we measure our performance using the "perfect predictions" made by the participants. (Note that this may be unfairly penalizing our algorithm as it may be finding relevant documents not found by the other systems.) We consider an annotation we make as correct if it exactly matches a "perfect prediction" made by another system. For example, for gene vhl in PubMed Id 12169961, a "perfect prediction" made by one of the participants annotates the gene with *transcription* GO:0006350 using the following passage in text as evidence: *VHL inhibits transcription elongation, mRNA stability, Sp1-related promoter activity and PKC activity*. For the same gene-article pair we consider our prediction to be correct if we find *transcription* GO::0006350 in exactly the same passage of text.

Since the target genes are human, and since mouse is a species with a genome similar to humans', for each target gene we compute the set $CSM(g, a)$ by searching in the articles for only the GO codes annotated to its mouse orthologous gene (except the GO codes marked with evidence codes *IEA* and *ISS* to avoid circular references). The orthologous databases we used are MGI and the part of SwissProt related to mouse genes[e].

For each human gene, we extract from MGI and SwissPro the GO annotations of the mouse gene with the same name as the target gene or with a name found in the Human-Mouse Orthology maps available from MGI[f]. We were able to find GO codes for about 61% of the human genes. For the genes whose orthologs had no GO annotations, we did not perform any search, so for these genes the sets $CSM(g, a)$ are empty. Next, for each gene g in article a we compute set $CSC(g, a)$ by searching in text for all possible 17,600 GO codes and eliminating illogical annotations using the χ^2 coefficient. Sets CSM and CSC are the union

[e]http://au.expasy.org/sprot/sprot-top.html, as of July 12, 2004.
[f]ftp://ftp.informatics.jax.org/pub/reports/index.html.

of all $CSM(g,a)$ and $CSC(g,a)$ over all genes and articles.

Table 1. Results on BioCreAtIve dataset.

System	Precision	TP (Recall)	F-measure
CSM	0.364	16 (0.068)	0.114
CSC	0.182	44 (0.185)	0.178
CSM + CSC	0.241	51 (0.215)	**0.227**
Ray and Craven[23]	0.213	52 (0.219)	0.216
Chiang and Yu[7,8]	0.327	37 (0.156)	0.211
Ehler and Ruch[11]	0.123	78 (0.329)	0.179
Couto et al.[10]	0.089	58 (0.245)	0.131
Verspoor et al.[26]	0.055	19 (0.080)	0.065
Rice et al.[25]	0.035	16 (0.068)	0.046

Table 1 compares the performance of our algorithms ($CSM, CSC, CSM + CSC$), with the performance of the participants in the competition as presented in Blaschke et al[3]. For each participant we report the best results they obtained in the competition. In the second column, TP stands for the number of true positive predictions made by a system (the number of predictions where both the protein and the GO code found in the passage are correct). Recall is computed as the ratio between TP and the total number of correct predictions (237).

In general, the results show the trade-off between precision and recall, and the systems that did well on precision obtained a recall much lower than the systems that did well on recall (which in turn obtained a lower precision). For example, Chiang and Yu's system has the best precision, 0.327, although the recall 0.156, is much lower than the best recall obtained by Ehler and Ruch's system, 0.329, which in turn had a lower precision, 0.123.

Although high precision is desirable, high recall is also important. For this reason, the F-measure (defined as the the harmonic mean of precision and recall) is considered a better metric for comparing results since a system has to maximize both precision and recall. The best F-measure is obtained by Ray and Craven's system, 0.216 with a precision of 0.213 and a recall of 0.219.

CSM obtains an F-measure of 0.114, although its precision, 0.364 is higher than any precision obtained in the competition. In turn, $CSM + CSC$ obtains an F-measure of 0.227, which is higher than the best F-measure in the competition 0.216, obtained by Ray and Craven's system.

CSC obtains an F-measure of 0.178. This result shows the effect of the CSC heuristic on our task but further analysis would needed to determine how often the co-occurring GO codes truly reflect logical or illogical combinations.

4.2. *Results on the EBI Human and MGI Mouse Datasets*

In this section we further compare the performance of our algorithms by evaluating them on much larger datasets and comparing them with the performance of Chiang and Yu's system[8], which performed well in the BioCreAtIve competition[g].

We present experimental results on two GO-annotated databases: EBI human and MGI Mouse (July 12 2004 versions). On each database, for every gene-document pair, we attempt to predict the manually annotated GO codes. Similarly to Chiang and Yu[8] we restrict our study to the genes we found in abstracts only, although curation is done on the full text.

The EBI human test set consisted of 4,410 genes annotated with 13,626 GO codes in 5,714 abstracts. The MGI test set consisted of 2,188 mouse genes annotated with 6,338 GO codes in 1,947 abstracts. For human genes, the orthologous databases we used are MGI and the part of SwissPro related to mouse genes. For mouse genes the orthologous databases are EBI human and the part of SwissPro related to human genes.

Table 2 shows the results obtained by CSM, $CSM + CSC$ and Chiang and Yu's system on both datasets. Chiang and Yu used the same data for both training and testing[8], which artificially inflates how well it would perform under real test conditions. In our case the test collection represents new data for our algorithm. While Chiang and Yu's algorithm generally achieves higher precision, $CSM + CSC$ obtains a better F-measure. On EBI human, Chiang and Yu obtain an F-measure of 0.105 while $CSM + CSC$ obtains 0.118. On MGI, Chiang and Yu obtain an F-measure of 0.089 while $CSM + CSC$ obtains 0.140.

Our experimental results also show that predicting molecular functions and cellular components may be easier than predicting biological processes. For example, on EBI, $CSM + CSC$ obtains an F-measure of 0.154 (for MFs), 0.124 (for CCs) and only 0.08 (for BPs). A possible explanation for this could be the fact that BPs have longer strings which are more difficult to match in text.

5. Conclusions

We propose a method that annotates genes with GO codes using the information available from other species [h]. In particular, we search in text for only the GO codes annotated to a gene that is an orthologous of the target gene. Since this

[g]These authors have made publicly available the annotations that their system assigns to *all* genes in LocusLink, http://gen.csie.ncku.edu.tw/meke3, as of September 2002. To obtain a fair comparison, our evaluation uses only the genes they annotate and only documents published before September 2002.

[h]Annotations and software available at http://biotext.berkeley.edu.

Table 2. Results on EBI human and MGI datasets.

Test set	System	Precision	Recall	F-score
EBI	CSM	0.289	0.033	0.060
	CSM + CSC	0.163	0.092	**0.118**
	Chiang and Yu	0.318	0.063	0.105
MGI	CSM	0.328	0.049	0.086
	CSM + CSC	0.168	0.121	**0.140**
	Chiang and Yu	0.332	0.051	0.089

approach results in low recall, we also search for all the GO codes in the Gene Ontology, but eliminate illogical annotations using the correlations between GO codes computed on the orthologous genes database. We test our algorithm on three collections: BioCreAtIve, EBI human and MGI. Experimental results show that our algorithm consistently achieves higher F-measure than other solutions.

In the future we plan to explore how to improve the performance of our system; one possibility is to combine or use a voting scheme to decide between the predictions made with our system and the predictions made using a machine learning algorithm like that of Ray and Craven[23]. In addition, we plan to investigate how effective using genes with sequences similar to the target gene (but not orthologous to the gene) is for predicting GO annotations.

Acknowledgements. This research was supported by NSF grant DBI-0317510 as well as a gift from Genentech.

References

1. G. Bhalotia, P.I. Nakov, A.S. Schwartz, and M.A. Hearst. Biotext team report for the trec 2003 genomic track. In *Proceedings of TREC 2003*, pages 612–621, 2003.
2. 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. In *Nucleic Acids Res*, volume 31, pages 193–195, 2003.
3. C. Blaschke, E. A. Leon, M. Krallinger, and A. Valencia. Evaluation of biocreative assessment of task 2. *BMC Bioinformatics*, 6(S1), 2005.
4. D.L. Brutlag. Genomics and computational molecular biology. *Current Opinion in Microbiology*, 1(3):340–345, 1998.
5. E. Camron, D. Barrell, V. Lee, E. Dimmer, and R. Apweiler. The gene ontology annotation (goa) database - an integrated resource of go annotations to the uniprot knowledgebase. 4(1):5–6, 2004.
6. F. Chalmel, A. Lardenois, J. D. Thompson, J. Muller, J.-A. Sahel, T. Lveillard, and O. Poch. Goanno: Go annotation based on multiple alignment. *Bioinformatics*, 19(11):1417–1422, 2005.
7. J. Chiang and H. Yu. Extracting functional annotations of proteins based on hybrid text mining approaches. In *Proc. of BioCreative Workshop*, 2004.
8. J.-H. Chiang and H.-C. Yu. Meke:discovering the functions of gene products from

biomedical literature via sentence alignment. *Bioinformatics*, 19(11):1417–1422, 2003.
9. The Gene Ontology Consortium. Gene ontology: tool for the unification of biology. *Nature Genet.*, 25(1):25–29, 2000.
10. F. M. Couto, M. J. Silva, and P. Coutinho. Figo: Finding go terms in unstructured text. In *Proc. of BioCreative Workshop*, 2004.
11. F. Ehler and P. Ruch. Preliminary report on the biocreative experiment. In *Proc. of BioCreative Workshop*, 2004.
12. M.B. Eisen, P.T. Spellman, P.O. Brown, and D. Botsein. Cluster analysis and display of genome-wide expression patterns. *Proc. Natl. Acad. Sci*, 95(25):14863–14868, 1998.
13. S. Hennig, D. Groth, and H. Lehrach. Automated gene ontology annotation for annonymous sequence data. *Nucleic Acids Res*, 31(13):3712–3715, 2003.
14. W.R. Hersh, R.T. Bhuptiraju, L. Ross, A.M. Cohen, and D.F. Kraemer. Trec 2004 genomics track overview. In *Proceedings of TREC 2004*, 2004.
15. L. Hirschman, A. Yeh, C. Blaschke, and A. Valencia. Overview of biocreative: critical assessment of information extraction for biology. *BMC Bioinformatics*, 6(S1), 2005.
16. V.R. Iyer, M.B. Eisen, D.T. Ross, G. Schuler, T. Moore, J.C. Lee, J.M. Trent, L.M. Staudt, J. Hudson, and M.S. Boguski M.S. et.al. The transcriptional program in the response of human fibroblasts to serum. *Science*, 283(5398):83–87, 1999.
17. L.J. Jensen, R. Gupta, H.H. Staerfeldt, and S. Brunak. Prediction of human protein function according to gene ontology categories. *Bioinformatics*, 19(5):635–642, 2003.
18. C.A. Joslyn, S.M. Mniszewski, A. Fulmer, and G. Heaton. The gene ontology categorizer. *Bioinformatics*, 4(20):1169–1177, 2004.
19. S. Khan, G. Situ, K. Decker, and C. J. Schmidt. Gofigure: Automated gene ontology annotation. *Bioinformatics*, 19(18):2484–2485, 2003.
20. O.D. King, R.E. Foulger, S.S. Dwight, J.V. White, and F.P. Roth. Predicting gene function from patterns of annotation. *Genome Research*, 13(5):896–904, 2003.
21. A. Koike, Y. Niwa, and T. Takagi. Automatic extraction of gene/protrin biological functions from biomediacal text. *Bioinformatics*, 21(7):1227–1236–1422, 2005.
22. C. Manning and H. Schutze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.
23. S. Ray and M. Craven. Learning statistical models for annotating proteins with function information using biomedical text. *BMC Bioinformatics*, 6(S1), 2005.
24. S. Raychaudhuri, J. T. Chang, P. D. Sutphin, and R. Altman. Associating genes with gene ontology codes using a maximum entropy analysis of biomedical literature. *Genome Research*, 12(1):203–214, 2002.
25. S. B. Rice, G. Nenadic, and B. J. Stapley. Protein function asignment using term-based support vector machines. In *Proc. of BioCreative Workshop*, 2004.
26. K. Verspoor, J. Cohn, C. Joslyn, and S. Mniszewski. Protein annotation as term categorization in the gene ontology. In *Proc. of BioCreative Workshop*, 2004.
27. A. Vinayagam, R. Koenig, J. Moormann, F. Schubert, R. Eils, K.H. Glatting, and S. Suhai. Applying support vector machines for gene ontology based gene function prediction. *BMC Bioinformatics*, 5(1), 2004.
28. H. Xie, A. Wasserman, Z. Levine, A. Novik, V. Grebinski, A. Shoshan, and L. Mintz. Large-scale protein annotation through gene ontology. *Genome Res*, 12(5):785–794, 2002.