Pacific Symposium on Biocomputing 12:221-232(2007)

ANNOTATING GENES USING TEXTUAL PATTERNS

ALI CAKMAK GULTEKIN OZSOYOGLU

Department of Electrical Engineering and Computer Science
Case Western Reserve University
Cleveland, OH 44106, USA
{ali.cakmak, tekin}@case.edu

Annotating genes with Gene Ontology (GO) terms is crucial for biologists to characterize the traits of genes in a standardized way. However, manual curation of textual data, the most reliable form of gene annotation by GO terms, requires significant amounts of human effort, is very costly, and cannot catch up with the rate of increase in biomedical publications. In this paper, we present GEANN, a system to automatically infer new GO annotations for genes from biomedical papers based on the evidence support linked to PubMed, a biological literature database of 14 million papers. GEANN (i) extracts from text significant terms and phrases associated with a GO term, (ii) based on the extracted terms, constructs textual extraction patterns with reliability scores for GO terms, (iii) expands the pattern set through "pattern crosswalks", (iv) employs semantic pattern matching, rather than syntactic pattern matching, which allows for the recognition of phrases with close meanings, and (iv) annotates genes based on the "quality" of the matched pattern to the genomic entity occurring in the text. On the average, in our experiments, GEANN has reached to the precision level of 78% at the 57% recall level.

1. Introduction

In this paper, we present GEANN (Gene Annotator), a system to automatically infer new Gene Ontology (GO) annotations for genes from biomedical papers based on the evidence support linked to PubMed, a biological literature database of 14 million papers. Currently, annotations for GO, a *controlled term vocabulary* describing the central attributes of genes [1], are most reliably done manually by experts who read the literature, and decide about appropriate annotations. This approach is slow and costly. And, compounding the problem is the rate of increase in the amount of available biological literature: at the present time, about 223,000 new genomics papers (that contain at least one of the words "gene", "protein" or "rna", and are published in 2005) per year are added to PubMed [3], far outstripping capabilities of a manual annotation effort. Hence, effective computational tools are needed to automate annotation of genes with GO terms.

Currently, possibly many genes without appropriate GO annotations exist even though there may be sufficient annotation evidence in a scientific paper. We have observed that, as of Jan. 2006, only a small portion of the papers in

PubMed has been referred to in support of gene annotations (i.e., 0.9% of 3 million PubMed genomics papers with abstracts). We give an example.

Example. The following is an excerpt from an abstract [18] which discusses experiments indicating the ***translation repressor activity*** (GO: 0030371) of the ***gene*** p97. However, presently *gene* p97 does not have the *translation repressor activity* annotation.

"...experiments show that p97 suppresses both cap-dependent and independent translation ... expression of p97 reduces overall protein synthesis...results suggest that p97 functions as a general repressor of translation by forming..." .

GEANN can be used to (i) discover new GO annotations for a gene, and/or (ii) increase the annotation strength of existing GO annotations by locating additional paper evidence. We are currently integrating GEANN into *PathCase* [2], a system of web-based tools for metabolic pathways, in order to allow users to discover new GO annotations. In general, GEANN is designed to:

- facilitate and expedite the curation process in GO, and
- extract explicit information about a gene that is implicitly present in text.

GEANN uses paper abstracts, and utilizes textual pattern extraction techniques to discover GO annotations automatically. GEANN's methodology is to *(i)* extract textual elements identifying a GO term, *(ii)* construct patterns with reliability scores, conveying the semantics of how confidently a pattern represents a GO term, *(iii)* extend the pattern set with longer ones via "crosswalks", *(iv)* apply semantic pattern matching techniques using WordNet, and (*v*) annotate genes based on the "quality" of the matched pattern to the genomic entity occurring in the text.

In experiments, GEANN produced, on average, 78% precision at 57% recall. This level of performance is significantly better than the existing systems described in the literature, and compared in section 5.2.3 and section 6.

Overview: The GEANN implementation has two phases, namely, the training and the annotation phases. The goal of the training phase is to construct a set of patterns that characterize a variety of indicators for the existence of a GO annotation. As the training data, annotation evidence papers [1] are used. The first step in the training phase is the *tagging of genes* in the papers. Then, *significant terms/phrases* that differentially appear in the training set are extracted. Next, patterns are constructed based on (i) the significant terms/phrases, and (ii) the terms surrounding significant terms. Finally, each pattern is assigned a reliability score.

The annotation discovery phase looks for possible matches to the patterns in paper abstracts. Next, GEANN computes a matching score which indicates the strength of the prediction. Finally, GEANN determines the gene to be associated with the pattern match. At the end, new annotation predictions are ordered by their scores, and presented to the user.

The extracted patterns are *flexible* in that they match to a set of phrases with close meanings. GEANN employs WordNet [5] to deduce the *semantic closeness* of words in patterns. WordNet is an online lexical reference system in which nouns, verbs, adjectives and adverbs are grouped into synonym sets, and these synonym sets are hierarchically organized through various relationships.

The paper is organized as follows. In section 2, we elaborate on *significant term discovery*, and *pattern construction*. Sections 3 and 4 discuss pattern matching and the scoring scheme, respectively. Section 5 summarizes the experimental results. In section 6 (and 5.2.3), we compare GEANN to other similar, competing, systems.

2. Pattern Construction

In GEANN, the identifying elements of a GO concept are the representations of the concept in textual data. And, the terms surrounding the identifying elements are considered as auxiliary descriptors of the GO concept. A pattern is an abstraction which encapsulates the identifying elements and the auxiliary descriptors together in a structured manner. More specifically, a pattern is organized as a 3-tuple: {LEFT} <MIDDLE> {RIGHT} where each element corresponds to a set (bag) of words. <MIDDLE> element is an ordered sequence of *significant terms* (*identifying elements*), {LEFT} and {RIGHT} elements correspond to word sets that appear around significant terms (*auxiliary descriptors*). The number of terms in the left and the right elements is adjusted by a *window* size.

Each word or phrase in the significant term set is assigned to be the middle element of a newly created *pattern template*. A pattern is an instance of a pattern template which may lead to several patterns with a common middle element, but (possibly) different left or right elements. We give an example.

Example. Two of the patterns that are created from the pattern template {LEFT} <*rna polymerase ii*> {RIGHT} are listed below where *rna polymerase ii* is found to be a significant term within the context of GO concept *positive transcription elongation factor* with the window size of three. {LEFT} and {RIGHT} tuples are instantiated from the surrounding words that appear before or after the significant term in the text.

{increase catalytic rate}<rna polymerase ii>{transcription suppressing transient}
{proteins regulation transcription}<rna polymerase ii>{initiated search proteins}

Patterns are contiguous blocks, that is, no space is allowed between the tuples in a pattern. Each tuple is a nag of words which are tokens delimited by white space characters. Since the stop words are eliminated in the preprocessing stage, the patterns do not include words like “the”, “of”, etc.

2.1. *Locating Significant Terms and Phrases*

Some words or phrases appearing frequently in the abstracts provide evidence for annotations by a specific GO term. For instance, *RNA polymerase II* which performs elongation of RNA in eukaryotes appears in almost all abstracts associated with the GO term "*positive transcription elongation factor activity*". Hence, intuitively, such *frequent term occurrences* should be marked as indicators of a possible annotation. In order to avoid marking word(s) common to almost all abstracts (*e.g.,* "cell"), the document frequency of a significant term is enforced to be below a certain threshold (10% in our case). The words that constitute the name of a GO term are by default considered as significant terms.

Frequent phrases are constructed out of *frequent terms* through a procedure similar to the Apriori algorithm [9]. First, individual frequent terms are obtained using the IDF (inverse document frequency [4]) indices. Then, frequent phrases are obtained by recursively combining individual frequent terms/phrases, provided that the constructed phrase is also frequent.

In order to obtain *significant terms*, one can use various methods from random-walk networks to correlation mining [9]. Since the training set for each GO term is most of the time not large, and to keep the methodology simple, we use frequency information to determine the significant terms.

2.2. *Pattern Crosswalks*

Extended patterns are constructed by virtually walking from one pattern to another. The goal is to create larger patterns that can eliminate false GO annotation predictions, and boost the true candidates. Based on the type of the walk, GEANN creates two different extended patterns: (i) *side-joined*, and (ii) *middle-joined* patterns.

Transitive Crosswalk: Given a pattern pair P_1 = {left1} <middle1> {right1}, and P_2 = {left2} <middle2> {right2}, if {right1} = {left 2}, then patterns P_1 and P_2 are merged into a 5-tuple *side-joined (SJ) pattern* P_3 = {left1} <middle1> {right1} <middle2> {right2}. Next, we give an example of a SJ pattern that is created for GO term *positive transcription elongation factor*.

Example. P_1 = {factor increase catalytic}<rate>{RNA polymerase II}
P_2 = {RNA polymerase II}<elongation factor>{[ge]}
[SJ *Pattern*] P_3 = {factor increase catalytic}<rate><RNA polymerase II>{elongation factor}{[ge]}

SJ patterns are helpful in detecting consecutive pattern matches that partially overlap in their matches. If there exist two consecutive regular pattern matches, then such a match should be evaluated differently than two separate matchings of regular patterns as it may provide a stronger evidence for the existence of a possible GO annotation in the match region. Note that pattern merging through crosswalks is performed among the patterns of the same GO concept.

Middle Crosswalk: Based on the partial overlapping between the middle and side (right or left) tuples of patterns, we construct the second type of extended patterns. Given the same pattern pair P_1 and P_2 as above, the patterns can be merged into a 4-tuple *middle-joined (MJ) pattern* if at least one of the following cases holds.

a. Right middle walk: {right1} ∩ <middle2> ≠ ∅ and <middle1> ∩ {left2}=∅

b. Left middle walk: <middle1> ∩ {left2} ≠ ∅ and {right1} ∩ <middle2>=∅

c. Middle walk: <middle1> ∩ {left2} ≠ ∅ and {right1} ∩ <middle2> ≠ ∅

MJ patterns have two middle tuples. For case (a), the first middle tuple is the intersection of {right1} and <middle2> tuples. Case (b) is handled similarly. As for case (c), the first and the second middle tuples are subsets of <middle1> and <middle2>. Below, we give an example of MJ pattern construction for the GO term *positive transcription elongation factor.*

Example. *(Middle-joined pattern construction)*

P_1 = {[ge] facilitates chromatin} <transcription> {chromatin-specific elongation factor}
P_2 = {classic inhibitor transcription} <elongation rna polymerase ii> {pol II}
[*MJ Pattern*] P_3 = {[ge] facilitates chromatin} <transcription> <elongation> {pol II}

Like SJ patterns, MJ patterns capture consecutive pattern matches in textual data. In particular, MJ patterns detect partial information that may not be recognized otherwise, since we enforce the full matching of middle tuple(s) to locate a pattern match, which is discussed next.

3. Handling Pattern Matches

Since middle tuples of a pattern are composed of significant terms, the condition for a pattern match is that the middle tuple of the pattern should be completely included in the text. For the matching of the left and the right tuples, GEANN employs *semantic matching*. We illustrate with an example.

Example. Given a pattern "*{increase catalytic rate}<transcription elongation>{RNA polymerase II}*", we want to be able to detect the phrases which give the sense that "transcription elongation" is positively affected. Through semantic matching, phrases like "*stimulates rate of transcription elongation*" or "*facilitates transcription elongation*" are also matched to the pattern.

GEANN first checks if an exact match is possible between the left/right tuples of the pattern, and the surrounding words of the matching phrase. Otherwise, GEANN employs WordNet [5] to check if they have similar meanings using an open source library [22] as access interface to WordNet. First, a semantic similarity matrix, `R[m,n]`, containing each pair of words is built, where R[i, j] is the semantic similarity between the most appropriate sense of the word at position i of phrase X, and the word at position j of phrase Y. The most appropriate sense of the word is found by through a sense disambiguation process. Given a word w, each sense of the word is compared against the senses of the surrounding words, and the sense of w with the highest similarity to the surrounding words is selected as the most appropriate sense. To compute semantic similarity, we adopt a simple approach: the semantic similarity between word senses w_1 and w_2 is inversely proportional to the length of the path between the senses in WordNet. The problem of computing semantic similarity between two sets of words X and Y is considered as the problem of computing a maximum total matching weight of a bipartite graph [7], where X and Y are two sets of disjoint nodes (*i.e.,* words in our case). The Hungarian Method [7] is used to solve this problem where R[i, j] is the weight of the edge from i to j. Finally, each individual pattern match is scored based on (i) the score of the pattern itself, and (ii) the semantic similarity computed using WordNet.

Having located a match, the next step is to decide on the gene that is associated to the match. To this end, two main issues are resolved: (i) detecting gene names in the text, and (ii) determining the gene to be annotated among possible candidates. For the first task, we utilized a decent biological *named entity tagger*, called Abner [20]. For the second task of locating the gene to be annotated, GEANN first looks into the sentence containing the match, and locates the genes that are positioned before/after the matching region in the sentence, or else in the previous sentence and so on. The confidence of the annotation decays as the distance from the gene to the matching phrase increases. For more details, please see [14].

4. Pattern Evaluation and Scoring

4.1. *Scoring Regular Patterns*

Each constructed pattern is assigned a score conveying the semantics of how confidently a pattern represents a GO term. GEANN uses several heuristics for the final score of a pattern based on the structural properties of its middle tuple.

i) Source of Middle Tuple [MT]*:* The patterns whose middle tuples fully consist of words from the GO term name gets higher score than those with middle tuples constructed from the frequent terms.

ii) Type of Individual Terms in the Middle Tuple [TT]*:* Contribution of each word from GO term name changes according to (a) the selectivity, *i.e.,* the occurrence frequency of the word among all GO term names, and (b) the position of the word in GO term name based on the observation that words in a GO term name get more specific from right to left [21].

iii) Frequency of the Phrase in the Middle Tuple [PC]*:* A pattern's score is inversely proportional to the frequency of the middle tuple throughout the papers in the database.
iv) Term-Wise Paper Frequency of the Middle Tuple [PP]*:* The patterns with middle tuples which are highly frequent in the GO term's paper set get higher scores.

Based on the reasoning summarized above, GEANN uses the following heuristic score function:

```
PatternScr = (MT + TT + PP) * Log(1/PC)
```

4.2. *Scoring Extended Patterns*

(a) Scoring SJ Patterns: SJ patterns serve for capturing consecutive pattern matches. Our scoring scheme differentiates between two-consecutive and two-single pattern matches where consecutive pattern matches contribute to the final score proportional to some exponent of the sum of the pattern scores (after experimenting with different values of exponents in the extended pattern score functions for the highest accuracy, for the experimental results section, j and k were set to 2 and 1.5, respectively). This way, GEANN can assign considerably higher scores to consecutive pattern matches which are considered as much stronger indicators for an annotation than two individual pattern matches.

$$Score(SJ\ Pattern) = (\ Score(Pattern1) + Score(Pattern2)\ )^{j}$$

(b) Scoring MJ Patterns: Consistent with the construction process, the score computation for MJ patterns is more complex in comparison to SJ patterns.

$$Score(Middle\text{-}joined\ Pattern) = (\ DegreeOfOverlap1 * Score(Pattern1)\ + DegreeOfOverlap2 * Score(Pattern2)\ )^{k}$$

where *DegreeOfOverlap* represents the proportion of the middle tuple of pattern1 (*pattern2*) that is included in the left tuple of pattern2 (*right tuple of pattern1*). In addition, GEANN considers the preservation of word order, represented by the *positionalDecayCoefficient*. The degree of overlap is computed by:

$$degreeOfOverlap = positionalDecayCoefficient\ * overlapFrequency$$

The positional decay coefficient is computed according to the alignment of the left or the right middle tuple of a pattern with the middle tuple of the other pattern. If a matching word is in the same position in both tuples, then the positional score of the word is 1, otherwise, it is 0.75.

$$PositionalDecayCoefficient = \frac{\sum_{w\ in\ Overlap} PosScore(w)}{Size(Overlap)}$$

5. Experimental Results

5.1. *Data Set*

In order to evaluate the performance of GEANN, we performed experiments on annotating genes in NCBI's Genbank with selected GO terms. A subset of PubMed abstracts was stored in a database. The experimental subset consisted of evidence papers cited by GO annotations, and reference papers that were cited for the gene maintained by GenBank. This corpus containing around 150,000 papers was used to approximate the word frequencies in the actual PubMed dataset. As part of pre-processing, abstracts/titles of papers were tokenized, stopwords were removed, and inverse document indices were constructed for each token. GEANN was evaluated on a set of 40 GO terms (24 terms from the biological process, 12 terms from mol. function, 4 term from cellular component subontology). Our decision on which terms to choose for the performance assessment is shaped by the choices made in two previous studies [16, 17] for comparison purposes. For a complete list of GO terms used in the experiments, see [14]. The evidence papers that are referenced from at least one of the test GO term are used for testing patterns. In total, 4694 evidence papers abstracts are used to to annotate 4982 genes where on the average each GO term has 120 evidence papers and 127 genes.

5.2. *Experiments*

Our experiments are based on the precision-recall analysis of the predicted annotation set. We use the k-fold cross validation scheme [9] (k=10 in our case). Precision is the ratio of the number of genes that are correctly predicted to the number of all genes predicted by GEANN. And, recall is the fraction of the correctly predicted genes in the whole set of genes that are known to be annotated with the GO term being studied. The genes that are annotated by GEANN, and yet, do not have a corresponding entry in Genbank are ignored as there is no way to check their correctness. Additionally, GEANN uses the following heuristics.

Heuristic 1 (Shared Gene Synonyms): *If at least one of the genes matching to the annotated symbol has the annotation with the target GO term, then this prediction is considered as a true positive.*

Heuristic 2 (Incorporating the GO Hierarchy): *A given GO term G also annotates all the genes that are annotated by any of its descendants (true-path rule).*

5.2.1. Overall Performance:

For this experiment, predicted annotations were ordered by their confidence scores. Precision and recall values were computed by considering top k predictions. k was increased by 1 at each step until either all the annotations for a GO term were located, or all the candidates in the predicted set were processed.

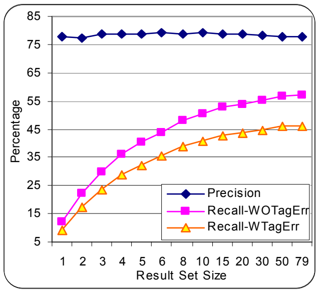

Figure 1: Overall System Performance & Approximate Error due to the NET

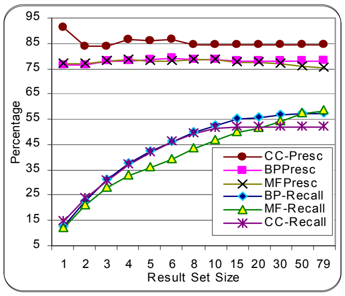

Figure 2: Annotation accuracy across different subontologies in GO

Observation 1: *From fig. 1, which presents the average precision/recall values, GEANN yields 78% precision (the top-most line) at 46% recall (the bottom-most line).*

The association of a pattern to a gene relies on the accurate tagging of genes in the text. However, named entity taggers (NETs) are still far from being perfect (ABNER has 77% recall, 68% precision). It may be quite difficult to exactly quantify NET errors. Thus, we took a minimalist approach, and attempted to compute the rate of error that is guaranteed to be due to the fault of the NET.

Heuristic 4 (Tagger Error Approximation): *If none of the synonyms of a gene has been recognized by the tagger in any of the papers which are associated with the target GO term G, then we label the gene as a tagger-missed gene.*

Observation 2**:** *After eliminating tagger-missed genes, the recall of GEANN increases to 57% from 46% at the precision level of 78% (the middle line in figure 1).*

Note that the actual error rate of the NET, in practice, may be much more than what is estimated above. In addition, eliminating tagger-missed genes does not affect the precision. Thus, precision is plotted only once.

5.2.2. Accuracy across Different Subontologies:

In experiment 2, the same steps of experiment 1 were repeated; but average accuracy values were computed within the individual subontologies. Figure 2 plots precision/recall values of different subontologies of GO (*MF*: Molecular Function, *BP:* Biological Process, *CC:* Cellular Component).

Observation 3: *GEANN has the best precision for CC where the precision reaches to 85% at 52% recall while MF yields the highest recall (58% at 75% precision).*

Observation 4: *CC almost always provides best precision values because the variety of the words to describe cellular locations may be much lower. However, CC has the lowest recall (52%) as the cellular location is well known for certain genomic entities, hence, are not stated explicitly in the text as much as MF or BP annotations.*

Observation 5: *Higher recall in MF is expected as, in general, the emphasis in a biomedical paper is on the functionality of a gene, where the process or the cellular location information is usually provided as secondary traits for the entity.*

5.2.3. Comparative Performance Analysis with Other Systems:

Raychaudhuri et al. [16] and Izumitani et al. [17] built paper classifiers to label the genes with GO terms through the classification of papers. Both works assume that a gene is a priori associated with several papers. This is a strong assumption in that if the experts are to invest sufficient time to read and associate a set of papers with a gene, then they can probably annotate the gene with the appropriate GO terms. Second, since both of the systems work at the document level, no direct evidence phrases are extracted from the text. Third, the classifiers employed by these studies need large training paper sets. In contrast, GEANN does not require a gene to be associated with any set of papers. Moreover, GEANN can also provide specific match phrases as evidence rather than the whole document. Fourth, GEANN handles the reconciliation of two different genomic databases whereas those studies have no such consideration. Izumitani et al. compares their system to Raychaudhuri et al.'s study for 12 GO terms. Our comparative analysis is also confined to this set of GO terms. Among these GO terms, five of them (Ion homeostasis, Membrane fusion, Metabolism, Sporulation) either have no or very few annotations in Genbank to perform 10-fold cross validation, and one of the test terms (Biogenesis) has recently became obsolete (i.e., removed from GO). Therefore, here we present comparative results for the remaining 6 GO terms. Table 1 provides the overall F-values [9] while Table 2 provides F-values in terms of the subontologies. F-value is a harmonic mean of precision and recall values, and computed as (2*Recall*Precision)/(Recall+Precision).

GO		**Izumitani**	**Raychaudhuri et al.**		
category	**GEANN**	**et al.**	**Top1**	**Top2**	**Top3**
GO:0006914	0.85	0.78	0.83	0.66	0.38
GO:0007155	0.66	0.51	0.19	0.19	0.13
GO:0007165	0.75	0.76	0.41	0.30	0.21
GO:0006950	0.69	0.65	0.41	0.27	0.24
GO:0006810	0.72	0.83	0.56	0.55	0.49
GO:0008219	0.75	0.58	0.07	0.06	0.02
Average	*0.74*	*0.69*	*0.40*	*0.33*	*0.25*

Table 1: Comparing F-Values against Izumitani and Raychaudhuri

GO Subontolgy	**GEANN**	**Izumitani et al.**
Biological Process	0.66	0.60
Molecular Function	0.66	0.72
Cellular Location	0.64	0.58
Average	*0.66*	*0.63*

Table 2: Comparing F-Values for GO Subontologies

Observation 6: *Although GEANN does not rely on the strong assumption that genes need to be associated with a set of papers, and provides annotation prediction at a finer granularity with much smaller training data, it is still comparable to or better than other systems in terms of accuracy.*

5.2.4. Contributions of Extended Patterns:

Finally, we evaluated the effects of extended patterns. The experiments were conducted by first utilizing extended patterns, and, then, without using extended patterns.

Observation 7: *The use of extended patterns improves the precision by as much as 6.3% (GO:0005198). However, as the average improvement is quite small (0.2 %), we conclude that the contribution of the extended patterns is unpredictable. We observe that extended patterns have a localized effect which does not necessarily apply in every case. Furthermore, since we only use paper abstracts, it is not very likely to find long descriptions that match to extended patterns.*

6. Related Work

The second task of the BioCreAtIvE challenge involves extracting the annotation phrases given a paper and a protein. Most of the evaluated systems had low precision (46% for the best performing system) [15]. We are planning to participate in this assesment challenge in the near future.

Raychaudhuri et al. [16] and Izumitani et al. [17] classify the documents, hence the genes that are associated to the documents into GO terms. As discussed above, even though GEANN is more flexible in terms of its assumptions, its performance is still comparable to these systems. Koike et al. [19] employs actor-object relationships from the NLP perspective. This system is optimized for the biological process subontology, and it requires human input and manually created patterns. Fleischman and Hovy [8] present a supervised learning method which is similar to our flexible pattern approach in that it uses WordNet. However, we use significant terms to construct additional patterns so that we can locate additional semantic structures while this paper only considers the target instance as the base of its patterns. Riloff [10] proposes a technique to extract the patterns. This technique ignores semantic side of the patterns. In addition, patterns are strict in that they require word-by-word exact matching. Brin's DIPRE [11] uses an initial set of seed elements as input, and uses the seed set to extract the patterns by analyzing the occurrences of seed instances in the web documents. SNOWBALL [12] extends DIPRE's pattern extraction system by introducing use of named-entity tags. Etzioni et al. developed a web information extraction system, KnowItAll [13], to automate the discovery of large collection of facts in web pages, which assumes redundancy of information on the web.

7. Conclusions and Future Work

In this paper, we have explored a new methodology to automatically infer new GO annotations for genes and gene products from biomedical paper abstracts. We have developed GEANN which utilizes existing annotation information to construct textual extraction patterns characterizing an annotation with a specific GO concept.

Exploring the accuracy of different semantic similarity measures for WordNet, disambiguation of genes that share a synonym, and determining scoring weight parameters experimentally are among the future tasks.

Acknowledgments

This research is supported in part by the NSF award DBI-0218061, a grant from the Charles B. Wang Foundation, and Microsoft equipment grant.

References

1. The Gene Ontology Consortium. The Gene Ontology (GO) database and informatics resource . Nucleic Acids Research 32, D258-D261, 2004
2. PathCase, available at http://nashua.case.edu/pathways
3. PubMed, available at http://www.ncbi.nlm.nih.gov/entrez/query.fcgi
4. Salton, G., Automatic Text Processing, Addison-Wesley, 1989.
5. Fellbaum, C. An Electronic Lexical Database. Cambridge, MA. MIT Press, 1998.
6. Mann, G. Fine-Grained Proper Noun Ontologies for Question Answering. SemaNet, 2002.
7. Lovasz, L. Matching Theory, North- Holland, New York, 1986.
8. Fleischman, M., Hovy, E. Fine Grained Classification of Named Entities. COLING 2002
9. Han, J., Kamber, M. Data Mining: Concepts and Techniques. The Morgan Kaufmann, 2000.
10. Riloff, E. Automatically Generating Extraction Patterns from Untagged Text. AAAI/IAAI,1996.
11. Brin, S. Extracting Patterns and Relations from the World Wide Web. WebDB 1998.
12. Agichtein, E., Gravano, L. Snowball: extracting relations from large plain-text collections.ACM DL 2000
13. Etzioni, O. et al. Web-scale information extraction in Knowitall: WWW 2004.
14. Extended version of the paper available at: http://cakmak.case.edu/TechReports/GEANN-Extended.pdf
15. Blaschke, C, Leon, EA, Krallinger M, Valencia A. Evaluation of BioCreAtIvE assessment of task 2. BMC Bioinformatics. 2005
16. Raychaudhuri, S. et al. Associating genes with Gene Ontology codes using a maximum entropy analysis of biomedical literature. Genome Res., 12(1):203–214.
17. Izumitani, T. et al. Assigning Gene Ontology Categories (GO) to Yeast Genes Using Text-Based Supervised Learning Methods. CSB 2004.
18. Imataka, H., Olsen, H., Sonenberg, N. A new translational regulator with homology to eukaryotic translation initiation factor 4G. EMBO J. 1997
19. Koike, A., Niwa, Y., Takagi, T. Automatic extraction of gene/protein biological functions from biomedical text. Bioinformatics 2005.
20. Settles, B. ABNER: An open source tool for automatically tagging genes, proteins, and other entity names in text. Bioinformatics, 2005.
21. Ogren, P. et al. The Compositional Structure of Gene Ontology Terms. PSB 2004.
22. WordNet Semantic Similarity Open Source Library http://www.codeproject.com/useritems/semanticsimilaritywordnet.asp