Gene Functional Annotation with Dynamic Hierarchical Classification Guided by Orthologs

Kazuhiro Seki, Yoshihiro Kino, and Kuniaki Uehara

Kobe University
1-1 Rokkodai, Nada, Kobe 657-8501, Japan
seki@cs.kobe-u.ac.jp

Abstract. This paper proposes an approach to automating Gene Ontology (GO) annotation in the framework of hierarchical classification that uses known, already annotated functions of the orthologs of a given gene. The proposed approach exploits such known functions as constraints and dynamically builds classifiers based on the training data available under the constraints. In addition, two unsupervised approaches are applied to complement the classification framework. The validity and effectiveness of the proposed approach are empirically demonstrated.

Key words: Gene ontology, String matching, Information retrieval

1 Introduction

Since the completion of the Human Genome Project, a large number of studies have been conducted to identify the roles of individual genes, which would help us understand critical mechanisms of human bodies, such as aging and disorders. The active research in the domain has been producing numerous publications. Although they are rich intellectual resources, it is extremely labor-intensive to collect all the information relevant to a given user information need, such as "a list of functions of gene X" or "a list of genes having function Y", since such information can be only accessed by extensive reading. To remedy the problem, numbers of organizations have been working to annotate each gene of model organisms with controlled vocabularies, called Gene Ontology (GO) terms, based on the contents of published scientific articles. GO is defined as a directed acyclic graph (DAG), and organized under three top level nodes: molecular function (MF), cellular component (CC), and biological process (BP). Currently, there are nearly 30,000 GO terms in total.

The effort of GO annotation has enabled uniform access to different model organism databases, including FlyBase, Mouse Genome Database (MGD), and Saccharomyces Genome Database, by the common vocabularies. However, the annotation requires trained human experts with extensive domain knowledge. With limited human resources and the ever-growing literature, it was reported that it would never be completed at the current rate of production [1].

Motivated by the background, this study proposes an approach to automatic GO annotation, which exploits the structure of GO and applies hierarchical

classification. In addition, we take advantage of orthologous genes and use their known gene functions as constraints to enable efficient learning. Moreover, we apply string matching-based and information retrieval model-based approaches to deal with the case where sufficient training data are not available.

2 Related Work

Due to the large number of genes, gene functions, and scientific articles, manual GO annotation is inevitably labor-intensive. In addition, because of the highly specialized contents, it requires skilled professionals with expertise in the domain. To alleviate the burden, TREC 2004 Genomics Track [2] and BioCreative [3] targeted automatic GO domain/term annotation.

The Genomics Track attempted to automate the process of assigning the first level of GO (i.e., MF, CC, BP), called "GO domains". The participants of the workshop were given a mouse gene and an article in which the gene appears and were expected to annotate zero to three GO domains with the gene based on the contents of the article. For this task, Seki and Mostafa [4] developed an approach featuring flexible gene mention extraction techniques based on a synonym dictionary and approximate name match. They used gene-centered representation by extracting fragments of an article mentioning the target gene and applied k nearest neighbor (kNN) classifiers with supervised term weighting.

In contrast to the Genomics Track only targeting GO domains, BioCreative aimed at assigning specific GO terms to human genes. Among others, Ray and Craven [5] looked at the occurrences of GO terms and their related terms to assign GO terms. Stoica and Hearst [6] took advantage of orthologs of a given gene and considered the GO terms already associated with them as candidates. Orthologs are genes in different species rooted from the same gene of their common ancestor and often have the same functions. Stoica and Hearst associated a given human gene with its mouse ortholog, and if the majority of terms consisting of each GO term assigned to the ortholog appeared in a given article, they assigned the GO term to the human gene. In addition, they used GO term co-annotation to prevent false positives. Their idea was based on the observation that there were cases where some GO terms were not usually co-annotated together to the same gene because annotating them together was illogical. For instance, "transcription (GO:0006350)" and "extracellular (GO:0005576)" are not likely to be co-annotated as transcription cannot happen outside of a cell.

Comparing the approaches taken at the Genomics Track and BioCreative, the participants for the former reported the effectiveness of supervised classification techniques. On the other hand, those for the latter mainly adopted string matching-based approaches. Such different strategies attributed to the fact that the former considers only three categories (i.e., GO domains), whereas the latter takes account of nearly 30,000 GO terms; dealing with less and general classes is more suitable for text categorization in terms of available training data and overfitting.

This study takes a classification approach to GO annotation by leveraging a limited amount of training data, where the GO structure and orthologous genes are used for guiding efficient classification. In addition, we complementarily use other unsupervised approaches when there is only insufficient training data so as to boost the coverage of GO annotation.

3 Proposed Approach

3.1 Overview

Our approach assigns appropriate GO terms for a given pair of gene g and an article d based on a set of text fragments mentioning g extracted from d. If there are multiple functions of g reported in d, we assign multiple GO terms corresponding to them. Roughly, our approach consists of the following steps: 1) Assign GO domains, 2) Obtain GO terms already assigned to the ortholog of the given gene g, 3) Assign GO terms by hierarchical classification, 4) Assign GO terms based on unsupervised approaches. Each step is described below.

3.2 Assigning GO domains

For GO domain annotation, we follow the approach proposed by Seki and Mostafa [4] who have reported the best performance in the literature. Simply put, for a given pair of gene g and article d, they first extract paragraphs mentioning g. Then, from the set of extracted paragraphs, a term vector is constructed to represent the input pair $\langle d, g \rangle$. Based on the representation, they assign GO domains by a variant of kNN.

3.3 Obtaining GO terms annotated with orthologs

After assigning GO domains, we identify promising GO term candidates in order to enable both effective and efficient GO term annotation. This study adapts the approach by Stoica and Hearst [6] using orthologs; That is, we consider only GO terms already assigned to the ortholog g' of a given gene g as GO term candidates. By this constraint, we can drastically reduce the number of GO terms to be considered from around 30,000 to only dozens at most. For instance, a mouse gene Sox21 has an ortholog in human genome, called SOX21, and the human gene has been already annotated with GO terms, including "RNA polymerase II transcription factor activity (GO:0003702)" and "establishment or maintenance of chromatin architecture (GO:0006325)", where the numbers in the parentheses are corresponding GO codes. Because these two genes are orthologous and are likely to have the same functions, we can expect higher precision by focusing only on these GO terms. Of course, it is also possible that true GO terms are not found in these GO term candidates. We will empirically investigate how often such cases occur in Section 4.2.

For the sources of the information regarding orthologs and their known gene functions (GO terms), this study uses two existing databases, MGD and Gene Ontology Annotation (GOA).

3.4 GO term annotation by dynamic hierarchical classification

Using the GO term candidates obtained through the ortholog of the given gene, we then assign specific GO terms by taking advantage of the structure of GO. For the above-mentioned example of Sox21, we consider only the GO terms already annotated with its ortholog as possible classes and train classifiers for them. However, as the number of the training instances with the classes (i.e., the GO terms) is often limited as discussed in Section 2, we enhance the training data set based on the GO structure. That is, for the candidate GO terms, we first identify their least common ancestor (LCA) and then train classifiers for the GO terms immediately under the LCA, where we consider only GO terms which have any candidates as descendants. For training data, we use not only the instances having the exact GO terms immediately under the LCA but also those having more specific GO terms under them. This way, one can use more training data and diminish the influence of the overfitting problem. Although this approach is similar to the hierarchical classification approach by McCallum et al. [7], a difference is that this study does not take into account all the classes in a given structure but only the limited number of the GO terms associated with a given gene through its ortholog. Also, training instances are dynamically harvested at each step of classification based on the GO term candidates, so as to learn classifiers on the fly.

A more precise algorithm of our dynamic hierarchical classification for GO term annotation is presented in Fig. 1, where the input is a test instance b, a set of training instances T, a set of GO term candidates C, and a set of GO domains assigned as described in Section 3.2; and the output is a set of GO terms F with which b is annotated. For each GO domain s, we identify GO term candidates C_s in the GO domain. If the number of the candidates $|C_s|$ equals 1, we unconditionally add the sole GO term candidate to the output F considering the fact that the GO domain s is already assigned and the GO term candidate is the only possible one to assign in the domain s. If $|C_s|$ is greater than 1, the following steps are carried out. First, we identify a set of GO terms C'_s immediately under the LCA and then, for each GO term in C'_s, we collect all the instances having GO terms under it. If the number of training instances for every GO term in C'_s is greater than a predefined threshold τ, we train classifier $\mathcal{F}$ and set per-class thresholds $\Theta = \theta_1, \ldots, \theta_{|C'_s|}$ to maximize F_1 score for each class $c'_i \in C'_s$ using the training instances. If classifier's output p_i for c'_i exceeds the threshold θ_i, c'_i is added to F in the case where c'_i is one of the GO term candidates, or we recursively apply the same procedure using c'_i as if it were a GO domain. If the number of training instances is below the threshold τ for any c'_i, we resort to the unsupervised approaches to avoid the overfitting problem as described next.

3.5 Unsupervised approaches to GO term annotation

In order to deal with the classes with insufficient training data (less than threshold τ), we make use of a string matching-based approach and an approach using

```
1  Input: test instance b, set of training instances T, set of GO term candidates
   C, set of predicted GO domains D;
2  Output: set of predicted GO terms F for b;
3  Variables: set of GO terms/domains S, prediction p_i ∈ ℝ for a GO term c'_i,
   threshold τ for training data size;

4  S = D
5  while S is not empty do
6    Take any GO term/domain out from S and set it to s
7    C_s = {c | c ∈ C under s}
8    if |C_s| = 1 then add C_s to F
9    else if |C_s| > 1 then
10     C'_s = {c' | GO terms immediately below s}
11     for each c' ∈ C'_s do
12       T_c' = {t | t ∈ T assigned any GO term under c'}
13     if ∀c', |T'_c| > τ then
14       Build a classifier F ↦ (p_1, ..., p_|C'_s|)
15       Determine per-class thresholds Θ = θ_1, ..., θ_|C'_s|
16       for each c'_i ∈ C'_s do
17         if p_i (predicted by F for b) > θ_i then
18           if c'_i ∈ C_s then add c'_i to F
19           else add c'_i to S
```

Fig. 1. Dynamic hierarchical GO term annotation algorithm.

an information retrieval model. These approaches were adapted from the related work in BioCreative and others.

String matching-based approach. Since GO terms are concise descriptions of gene functions in natural language, if a text contains a certain GO term, the text may be describing the corresponding gene function. This is not necessarily the case for general GO terms located at the higher level of the GO tree, such as "behaviour (GO:0007610)", but is likely to apply to more specific ones, such as "regulation of transcription from RNA polymerase II promoter (GO:0006357)". In this study, we use the edit distance to deal with some writing variations and differences. The edit distance basically counts the number of edit operations (i.e., insert, delete, and substitution) to convert a string (i.e., GO term) to another string (i.e., actual expression found in text). Also, to consider the different importance of words, we define different penalty costs for different words based on document frequencies (DF). We define the DF of a word w as the logarithm of the total number of GO terms containing w.

Information retrieval model-based approach. Another unsupervised approach has been proposed by Ruch [8]. We take a similar approach as him and assign GO terms based on a vector space model. Simply put, this approach measures the cosine similarity between a GO term and text and assigns the GO term if the similarity between them exceeds a predefined threshold. Essentially, this approach is similar to the string matching-based approach above except that this approach is less restrictive, not considering word orders.

4 Evaluation

4.1 Experimental settings

For evaluation, we use the data set provided for the TREC 2004 Genomics Track supplemented by GO term information. The data set consists of 849 training instances and 604 test instances, where each instance is a triplet of an article d represented by PubMed ID, a gene g mentioned in d, and a GO term f which is reported in d as a function of g. This data set is a subset of MGD, and thus, only dealing with mouse genes.

As an evaluation metric, we use F_1 score for direct comparison with the previous work, i.e., Genomics Track and BioCreative which used the same metric. F_1 is defined as a harmonic mean of recall (R) and precision (P). P is defined as the number of correct GO terms assigned divided by the number of GO terms assigned, and R is the number of correct GO terms assigned divided by the number of GO terms in the test data.

The proposed GO term annotation framework is general and by design does not depend on a particular classifier. Although the following experiments used kNN as it has been shown effective in the related work [4], it can be easily replaced with other classifiers.

4.2 Validity of the use of orthologs for GO annotation

As orthologs, we experimentally chose human and rat genes to annotate mouse genes. Our first experiment examined the validity of the use of those orthologs for GO term annotation. To be precise, we simply annotated input mouse genes with all the GO term candidates obtained from their orthologs *without* classification. This experiment reveals the coverage of the GO term candidates obtained through different species.

When comparing two species, human and rat, the latter works better for all of recall (0.800), precision (0.045), and F_1 (0.086). This is expected, as rat is genetically closer to mouse than human. Using rat genes, the recall was found 0.800, which means that 80.0% of true GO terms annotated to the test data are found in the GO terms already assigned to the rat orthologs. Differently put, this is the upper bound of recall for our framework to look only at GO term candidates obtained from orthologs. In this study, we focus on the 80.0% and recovering the remaining 20.0% are left for the future work.

4.3 GO term annotation by hierarchical classification

Table 1 shows the results for GO term annotation when our proposed approach (denoted as "Hierarchical") based on hierarchical classification was applied, where we used only rat genes as orthologs based on the observation in Section 4.2. In addition, the table shows for reference the results reported by Stoica and Hearst [6] and Chiang and Yu [9] on the BioCreative data set. Also, the results for standard flat classification without considering GO structure is included

Table 1. Comparison of the performance of GO term annotation.

Approaches	Precision	Recall	F_1
Stoica & Hearst [6]	0.168	0.121	0.140
Chiang & Yu [9]	0.332	0.051	0.089
Hierarchical (proposed approach)	0.248	0.210	0.227
Flat	0.041	0.551	0.075

Table 2. Results of GO annotation when hierarchical classification and unsupervised approaches are combined.

Approaches	Precision	Recall	F_1
Hierarchical	0.248	0.210	0.227
Hierarchical + Edit	0.238	0.245	0.242
Hierarchical + IR	**0.256**	0.275	**0.265**
Hierarchical + Edit + IR	0.236	**0.282**	0.257

(denoted as "Flat"). Note that "Flat" also looked at only GO term candidates obtained from orthologs and thus can be used to evaluate the effect of the use of the GO structure.

Comparing with the results by Stoica and Hearst [6] and Chiang and Yu [9], our proposed approach obtained the best performance in F_1. This result indicates that, if we can restrict the number of GO terms to be considered, supervised classification approaches can be effective even for GO term annotation for which a large number of classes otherwise exist. In addition, we can observe that the flat classification produced poor performance, which means that it is not sufficient only to restrict the number of possible classes.

4.4 GO term annotation by unsupervised approaches

In general, classification performance in precision improves up to some point as the training data size increases. However, because the GO terms with a large number of instances are limited, recall inevitably decreases with higher τ, the threshold for the number of instances. To improve recall, we apply two unsupervised approaches described in Section 3.5 when there are insufficient training data (less than τ). The results are shown in Table 2, where "Hierarchical" is the result by hierarchical classification taken from Table 1, "Edit" and "IR" denote approaches based on string matching and the IR model, respectively.

When Hierarchical is combined with one of Edit and IR, recall improved to 0.245 (+16.7%) and 0.275 (+31.0%), respectively. This result confirms the effectiveness of the unsupervised approaches and indicates that they work complementarily with our hierarchical classification approach. Especially, the IR approach resulted in a significant boost in recall and even improved precision as compared with Hierarchical alone. In addition, when both Edit and IR are combined with Hierarchical, recall further improved to 0.282 (+34.3%), which,

however, decreased precision. (It is attributed to the different value of τ used, which was chosen to maximize F_1 for each configuration.) Focusing on F_1, Hierarchical+IR was the best combination achieving an F_1 of 0.265. For comparison, when only IR without classification was applied for GO term annotation, F_1 was found to be around 0.150 (not shown in the table). This result also confirms that combining classification and unsupervised approaches is effective for GO term annotation.

5 Conclusions

This study proposed an approach to GO term annotation using orthologs to effectively guide hierarchical classification. In addition, two unsupervised approaches were applied when sufficient training data were not available. From the experiments on the Genomics Track data, we observed that 1) by using rat genes as orthologs, up to 80% of correct GO terms can be annotated; 2) using the GO term candidates obtained from orthologs, our hierarchical classifiers were able to annotate mouse genes at an F_1 of 0.227; and 3) by combining the hierarchical classification and the IR model-based approach, the performance improved up to 0.265. For future work, we aim to recover the remaining 20% of true GO terms not covered by the ortholog-based framework. This could be partly done by exploiting other homologs, e.g., paralogous and xenologous genes.

Acknowledgments. This work is partially supported by KAKENHI #21700169.

References

1. W.A. Jr. Baumgartner, K.B. Cohen, L.M. Fox, G. Acquaah-Mensah, and L. Hunter. Manual curation is not sufficient for annotation of genomic databases. *Bioinformatics*, 23(13):i41–48, 2007.
2. W. Hersh, R.T. Bhuptiraju, L. Ross, A.M. Cohen, and D.F. Kraemer. TREC 2004 genomics track overview. In *Proc. of the 13th Text REtrieval Conference*, 2004.
3. C. Blaschke, E. Leon, M. Krallinger, and A. Valencia. Evaluation of BioCreAtIvE assessment of task 2. *BMC Bioinformatics*, 6(Suppl 1):S16, 2005.
4. K. Seki and J. Mostafa. Gene ontology annotation as text categorization: An empirical study. *Information Processing & Management*, 44(5):1754–1770, 2008.
5. S. Ray and M. Craven. Learning statistical models for annotating proteins with function information using biomedical text. *BMC Bioinformatics*, 6(Suppl 1):S18, 2005.
6. E. Stoica and M. Hearst. Predicting gene functions from text using a cross-species approach. In *Proc. of the Pacific Symposium on Biocomputing*, pages 88–99, 2006.
7. A. McCallum, R. Rosenfeld, T.M. Mitchell, and A.Y. Ng. Improving text classification by shrinkage in a hierarchy of classes. In *Proc. of the 15th International Conference on Machine Learning*, pages 359–367, 1998.
8. P. Ruch. Automatic assignment of biomedical categories: toward a generic approach. *Bioinformatics*, 22(6):658–664, 2006.
9. J. Chiang and H. Yu. Extracting functional annotations of proteins based on hybrid text mining approaches. In *Proc. of the BioCreAtIvE*, 2004.