Identifying relevant full-text articles for GO annotation without MeSH terms

Chih Lee, Wen-Juan Hou, and Hsin-Hsi Chen

Department of Computer Science and Information Engineering
National Taiwan University
1, Section 4, Roosevelt Road,
Taipei, Taiwan, 106

E-mail: {clee, wjhou }@nlg.csie.ntu.edu.tw;
hh_chen@csie.ntu.edu.tw

Abstract

Gene Ontology (GO) is a controlled vocabulary. Given a gene product, GO enables scientists to clearly and unambiguously describe specific molecular functions of the gene product, specific biological processes in which it is involved, and specific cellular components to which it is localized. In this paper, we present our approach to identifying which papers have experimental evidence warranting annotation with GO codes. The training data set contains 375 relevant full-text articles and 5,462 irrelevant ones, and the test data set contains 420 positive full-text articles and 5,623 negative ones. We regarded this problem as a binary classification problem, and employed Support Vector Machines (SVMs) to distinguish positive articles from negative ones. Title, abstract, figure/table captions, and three standard sections – Results, Discussion, and Conclusion were the targets of feature extraction. Without incorporating MeSH (Medical Subject Headings) terms as part of the features, our system achieved *0.381* in Normalized Utility measure.

1 Introduction

Gene Ontology (GO) [1] is a system of keywords hierarchically organized as a directed acyclic graph with three main categories – biological process, cellular component, and molecular function. It provides a unified set of terms for the annotation of gene products in different organisms. The assignment of a GO term requires supporting evidence. The main source of evidence comes from published biomedical articles which contain accurate and reliable experimental results. Usually, curators have to read newly published papers to update their databases, and obviously they can hardly keep up with the rapidly growing number of new biomedical papers.

Efforts have been made to automatically annotate proteins with GO terms based on analysis of biomedical literature [6, 7, 10]. None of these works, however, exploited full-text articles, which have been shown to contain more information than abstracts [8]. The BioCreative workshop 2004 [16] initiated a GO annotation task, and provided 636 full-text documents from the Journal of Biological Chemistry. The evaluation was manually done by curators, and the overall results showed low performances which indicated a long way from practical application. Therefore, the categorization task in TREC 2004 genomics track [13] was simplified and limited to assignment of the three main categories. Also, full-text documents from three journals over two years were provided in this task.

The categorization task tried to mimic two of the classification activities carried out by human annotators in the mouse genome informatics (MGI) [15] system: a triage task and two variants of MGI's annotation task. Curators at MGI employ a three-step process to identify the papers most likely to describe gene function. First, articles from several hundred journals are

searched for keywords *mouse, mice, or murine*. Second, confronted with articles from the first step, curators determine which articles should be sent for curation. The goal for this triage process is to limit the number of articles for more exhaustive analysis. Finally, curators identify genes for which there are experimental evidence supporting assignment of GO terms.

Because of limited resources and time constraints, we did only the triage subtask. The goal of this task is to correctly identify which papers have been deemed to have experimental evidence warranting annotation of certain GO codes. Since this task can readily be regarded as a binary classification problem, we employed Support Vector Machines (SVMs) [9], which are especially suitable for binary classification problems.

Feature extraction is the key to successful classification in the machine learning approach, and is even more important than the underlying classification algorithm. When it comes to text categorization, the simple bag-of-words representation is often the first choice. However, the performance of cross validation on this simple approach was only about *0.1* in normalized utility (NU) measure [13]. Therefore, we tried some other representations and settled down to the one adopted in this paper. First, we obtained a list of GO terms [11] annotated to MGI markers. Then, a document was represented by the similarities to those GO terms. The details of this approach are discussed in the Section 3.

The rest of this paper is organized as follows. Section 2 presents the overview of our architecture. The basic idea and the experimental methods in this study are introduced in Section 3. Section 4 shows the results and makes some discussions. Finally, Section 5 concludes the remarks and suggests some future direction.

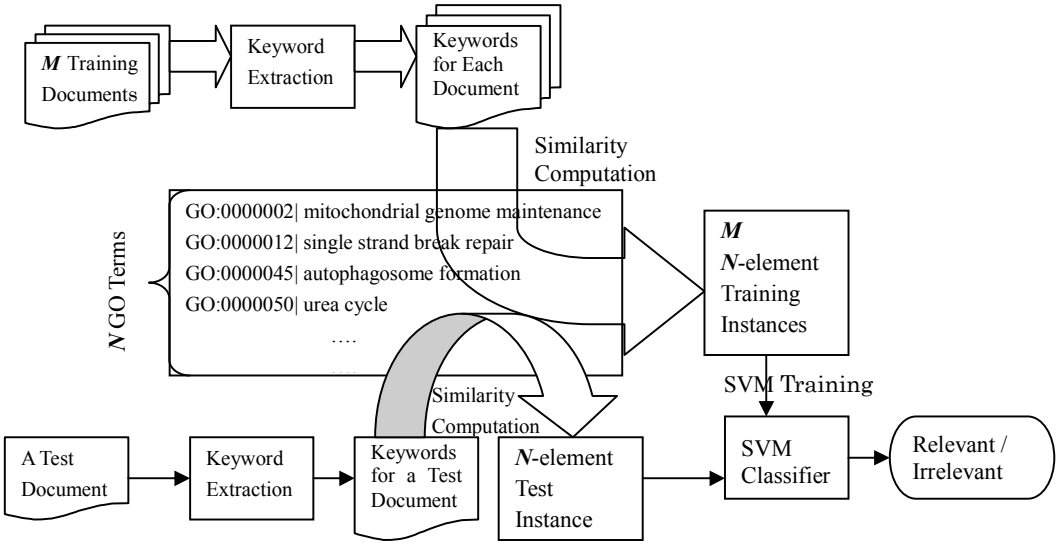

Figure 1: System overview.

2 Architecture Overview

Figure 1 shows the overall architecture of our method for the triage task. First, we obtained a list of *N* GO terms from the MGI website [11]. With *M* training documents available, we extracted a list of keywords for each document. Each document was then represented by the vector sum of similarity between each keyword and the list of *N* GO terms, forming an *N*-element vector. Detail of the conversion process is explained in Section 3. A SVM classifier was trained with the resulting *M N*-element vectors. Given a test document, it was converted to an *N*-element vector through the same process performed on those training documents. Afterwards, this test instance

was sent to the trained SVM classifier to decide whether it is relevant or not.

3 Methods

3.1 Document Preprocessing

Before performing classification, two document preprocessing operations were performed to extract more information from the full-text documents. The two operations were (1) acronym expansion and (2) keyword extraction.

- *Acronym expansion*

Once the combination of sections was decided, which is explained in Section 3.3, an operation was performed to substitute the long forms for the tagged acronyms, each of which referred to a glossary entry in the document. The reason for this operation is that acronyms are sometimes ambiguous, and their long forms obviously carry more information. An example of this operation is shown in Figure 2. In Figure 2, an abbreviation "IP_3" will be replaced with "inositol trisphosphase (IP_3)".

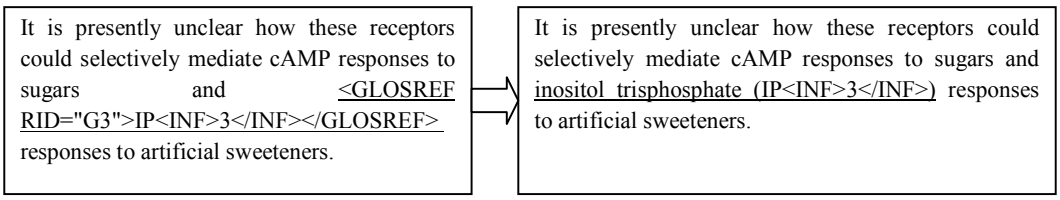

Figure 2: An example of acronym operation.

- *Keyword extraction*

After the acronym operation, the remaining SGML tags were removed from the documents for later keyword extraction. With the 554K-entry inflection table found in UMLS Knowledge Sources [5], the keywords were normalized and extracted from each document. In this step, only words that occur in both the inflection table and the document were extracted and normalized. The normalization here refers to the transformation of words to their root forms. For example, a verb in the past tense like "demonstrated" is normalized to its base form "demonstrate". Moreover, a plural noun like "receptors" is normalized to its singular form "receptor". Then, stop words were removed in the next stage. An example of keyword extraction is shown in Figure 3. The upper left part of Figure 3 contains the target document for keyword extraction. The lower left part illustrates the inflection table found in UMLS. The right part shows the extracted and normalized list of keywords.

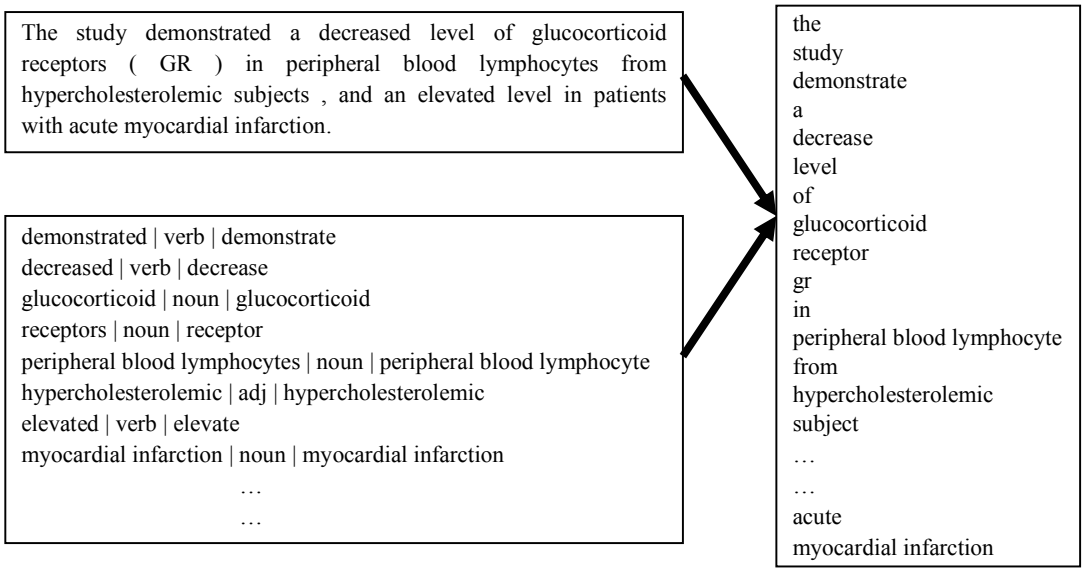

Figure 3: An example of keyword extraction and normalization.

3.2 Feature Extraction

Under the bag-of-words representation, the feature vector of an article was 23K long and the resulting dataset severely suffered from the data sparseness problem. Therefore, we attempted to solve this problem by reducing the dimension of feature vector via a list of N GO terms. Given a keyword, the similarity vector $\mathbf{V}_k$ for this keyword is the similarity between this keyword and the N GO terms. The feature vector $\mathbf{V}_D$ for a document is therefore the vector sum of the similarity vectors of all keywords it contains. In our study, Classic Dice (CD) coefficient was adopted as the similarity measure, and stop words were ignored at this stage. The formulas of computing the CD coefficient and similarity values are listed below.

$CD(A,B) = (2 \times Z)/(X + Y)$, where A, B are two strings, X is the number of tokens in A, Y is the number of tokens in B, and Z is the number of tokens occurring in both A and B.

$\mathbf{V}_k = \mathbf{Sim}(w_k, \mathbf{T}) = [CD(w_k, t_1)\ CD(w_k, t_2)\ ...\ CD(w_k, t_N)]^t$, where w_k is a keyword and $\mathbf{T}$ is the vector of N GO terms, t_1, t_2 and t_N is the first, second and Nth GO term, respectively.

$\mathbf{V}_D = \sum_{k \in D} \mathbf{V}_k$, where D is a document.

3.3 Exploitation of Full Text Documents

Which sections of an article should be the targets of feature extraction is also an important issue. In other words, we have to find out where the experimental evidence warranting annotation with GO codes resides in a document. The triage subtask is very much similar to the task 1 in KDD Cup 2002 [14], part of whose goal is to retrieve papers meeting the Flybase [12] gene-expression curation criteria. It was found in this competition that besides the title and abstract, much of the experimental evidence is contained in the figure captions. Hence, we started from the

combination of the title, abstract, figure captions and table captions as the base combination, and gradually included some other sections of the article to the base combination. Two other combinations were therefore constructed. Unfortunately, none of them outperformed the base combination under the aforementioned feature extraction method. These two combinations are briefly described below.

Some types of documents do not have the abstract part, and hence in this case, the body of the article is added to the simple combination, forming the second combination. In other words, if the abstract is absent, the body of the article is added to form the second combination. For the third one, if the abstract is present, the result, discussion, and conclusion sections are included. While these sections are intuitively evidence-rich sections, the third combination did not stand out as expected.

3.4 SVM Classification

The software package LIBSVM [4] was employed to deal with SVM-related operations. Radial basis function was adopted as the kernel function, and 4-fold cross validation was performed to select the model attaining the highest normalized utility, i.e., the best-performing parameters – *C* and *gamma*, which are the penalty constant in optimization and the parameter for radial basis kernel, respectively. Under our feature extraction method, the selected values for *C* and *gamma* are around 64 and 3.81^{-6}. Another issue worth addressing is the imbalance among the number of training and test examples. For the training data, there are 375 positive examples and 5462 negative examples, the ratio of which is about 1 to 20. Therefore, we tried to put more weights on the positive examples, i.e., the positive examples received larger *C* in SVM training. As expected, setting $C_{positive}$ to $20C_{negative}$ achieved the best performance in our cross validation.

3.5 Normalization versus Stemming

Due to time constrains, some of the methods simply followed our intuition without further verification or experience backup. The one in which we were interested the most is using normalization instead of stemming, which is a usual preprocessing operation in text categorization. Unlike stemming, normalization is more precise because it converts words to their base forms without losing too much information. As expected, further experiment displayed a 0.03 NU drop in the stemmed version.

4 Results and Discussions

Table 1 lists the results of our three official runs, the results of two other top-performing teams, and the results of the median-performing run and the worst-performing run. As mentioned in Section 3.3, NTU2v3N1 used the base combination, NTU3v3N1 used the second combination, and NTU4v3N1416 used the third combination. It seems that adding other sections besides the title, abstract, and captions introduced more noise and less useful information. The results may be explained by Schuemie *et al.*'s finding [8] that the information density is higher in the abstract than in all the other four standard sections – Introduction, Methods, Results and Discussion. Therefore, some filtering techniques should be applied to these four sections to remove non-informative and noisy contents.

The official run dimacsTfl9d was produced by Dayanik *et al.*'s system [2] which attained the best performance. Besides the title and abstract of an article, they used the MeSH terms attached to the article as the target of feature extraction. Bayesian logistic regression was adopted to perform classification. They also performed an interesting experiment which depended only on

the MeSH term “Mice” to make the decision, and found that using this term alone can outperform all the other systems. Fujita’s system [3] achieved slightly lower performance of 0.5494 NU (pllsgen4t3). They used terms from full text, gene entities and MeSH terms as the targets of feature extraction, and used SVM as the classifier. It is obvious that MeSH terms played an important role in distinguishing positive documents from negative ones, especially the term “Mice”. Since this task aimed to assist curators at MGI, it is reasonable that articles attached with the MeSH term “Mice” are very likely to be positive. Therefore, we can ascribe the high performance of these two systems to the use of MeSH terms.

As there are 375/420 positive examples and 5462/5623 negative examples in the training/test dataset, the curators at MGI will have to read about 15 papers to find a positive one if they do not get any hints in advance. For the best official run (dimacsTfl9d), curators will have to read roughly 6 papers to find one useful, and around 88 percent of the positive papers can be retrieved. Using our approach, curators will have to read roughly 10 papers to find one useful and only 69 percent of the positive papers can be retrieved. To put it in another way, the best run reduced from 15 to 6 the number of papers that the curators have to read to get a positive one, losing 12 percent of the useful papers. In our opinion, the best official run greatly alleviated the burden of curators, and our approach didn’t seem to help a lot. However, it is possible to combine our approach with others, making the filtering job even more effective.

Table 1: Results of official runs in the triage task.

	Normalized Utility	F-score	Recall	Precision
dimacsTfl9d	**0.6512**	0.2681	0.8881	0.1579
“Mice” run	**0.6404**	0.2572	0.8929	0.1502
pllsgen4t3	**0.5494**	0.2496	0.769	0.149
NTU2v3N1	**0.3810**	0.1752	0.6905	0.1003
NTU3v3N1	**0.3601**	0.1673	0.6857	0.0953
Median	**0.3425**	0.2345	0.469	0.1563
NTU4v3N1416	**0.3323**	0.1650	0.6357	0.0948
Worst	**0.0114**	0.0267	0.0143	0.2

5 Conclusion

We demonstrate our approach based on a list of GO terms in this paper. We tried three combinations of sections in an article as the target of feature extraction, and found the simplest one most useful. We hypothesize that filtering should be applied to sections other than Abstract before they can be used for feature extraction. Also, we found that normalization is a better preprocessing operation than stemming under our feature extraction approach.

Without the use of MeSH terms, our system performed slightly better than the median-performing system. With our approach, 69 percent of positive papers were retained at the precision rate of 10 percent. Although we didn’t achieve the best performance, it is possible to incorporate other ideas into our method, and combine other types of features with the existing ones.

References

[1] Ashburner, M., Ball, C.A., Blake, J.A., Botstein, D., Butler, H., Cherry, J.M. *et al*., Gene Ontology: Tool for the Unification of Biology, *Nature Genetics*, 25:25-29, 2000.

[2] Dayanik, A., Fradkin, D., Genkin, A., Kantor, P., Lewis, D.D., Madigan, D. and Menkov, V., DIMACS at the TREC 2004 Genomics Track (DRAFT), *TREC 2004 Notebook*, 162-171, 2004.

[3] Fujita, S., Revisiting Again Document Length Hypotheses TREC-2004 Genomics Track Experiments at Patolis, *TREC 2004 Notebook*, 152-161, 2004.

[4] Hsu, C.W., Chang, C.C. and Lin, C.J., A Practical Guide to Support Vector Classification, http://www.csie.ntu.edu.tw/~cjlin/libsvm/index.html, 2003.

[5] Humphreys, B.L., Lindberg, D.A., Schoolman, H.M. and Barnett, G.O., The Unified Medical Language System: an Informatics Research Collaboration, *Journal of American Medical Information Association*, 5(1):1-11, 1998.

[6] Pouliot, Y., Gao, J., Su, Q.J., Liu, G.G. and Ling, X.B., DIAN: a novel algorithm for genome ontological classification, *Genome Research*, 11:1766-1776, 2001.

[7] Prérez, A.J., Perez-Iratxeta, C., Bork P., Thode, G. and Andrade, M.A., Gene annotation from scientific literature using mappings between keyword systems, *Journal of Bioinformatics*, 20(13):2084-2091, 2004.

[8] Schuemie, M.J., Weeber, M., Schijvenaars, B.J.A., van Mulligen, E.M., van der Eijk, C.C., Jelier, R., Mons, B. And Kors, J.A., Distribution of information in biomedical abstracts and full-text publications, *Journal of Bioinformatics*, 20(16):2597-2604, 2004.

[9] Vapnik, V., *The Nature of Statistical Learning Theory*, Springer-Verlag, 1995.

[10] Xie, H., Wasserman, A., Levine, Z., Novik, A., Grebinskiy, V., Shoshan, A. And Mintz, L., Large-scale protein annotation through gene ontology, *Genome Research*, 12:785-794, 2002.

[11] ftp://ftp.informatics.jax.org/pub/reports/go_terms.mgi

[12] http://flybase.net

[13] http://medir.ohsu.edu/~genomics/2004protocol.html

[14] http://www.biostat.wisc.edu/~craven/kddcup/index.html

[15] http://www.informatics.jax.org

[16] http://www.pdg.cnb.uam.es/BioLink/workshop_BioCreative_04/