A Bayesian Integration Model for Improved Gene Functional Inference from Heterogeneous Data Sources

Noor Alaydie
Dept. of Computer Science
Wayne State University
Detroit, MI 48202
alaydie@wayne.edu

Chandan K. Reddy
Dept. of Computer Science
Wayne State University
Detroit, MI 48202
reddy@cs.wayne.edu

Farshad Fotouhi
College of Engineering
Wayne State University
Detroit, MI 48202
fotouhi@wayne.edu

ABSTRACT

Increasing amounts of biological data from various sources are being made available by high-throughput genomic technologies. However, no single biological data source analysis can fully unravel the complexities of the hierarchical gene function prediction. Therefore, the integration of multiple data sources is required to acquire a more precise understanding of the role of genes in the living organisms. In this paper, we develop a **Hi**erarchical **B**ayesian **iN**tegration algorithm, *HiBiN*, a general framework that uses Bayesian reasoning to integrate heterogeneous data sources for accurate gene function prediction. The system uses posterior probabilities to assign class memberships to samples using multiple data sources while maintaining the hierarchical constraint that governs the annotation of genes. We demonstrate that the integration of the diverse datasets significantly improves the classification quality for hierarchical gene function prediction in terms of several measures, compared to single-source prediction models and fused-flat model, which are the baselines we compared against.

Categories and Subject Descriptors

J.3.1 [**Computer Applications**]: Life and Medical Science - biology and genetics; I.5.2 [**Design Methodology**]: [Classifier design and evaluation].

General Terms

Bioinformatics, Machine Learning, Algorithms.

Keywords

Data integration, gene function prediction, hierarchical multi-label classification.

1. INTRODUCTION

Diverse high-throughput genomic data are becoming widely available. Since the functions of a significant number of genes are still unknown, such high throughput data can play a vital role in assigning accurate functional annotations on large-scale through computational prediction. Providing accurate predictions can advance experimental studies by providing specific hypothesis for targeted experimental testing [13]. Learning reliable classification models from a single dataset is often hard due to several complex issues including noise in the data, low relevance of the dataset for some functional classes and an insufficient number of training examples for building accurate classification models [9].

Several issues have contributed to the complexity of the gene functional classification problem. First, a gene may have multiple class labels. The functional classes are related to each other in a tree or a graph structure (Gene Ontology(GO) [4] or MIPS's FunCat taxonomy [11]). This leads to the second challenge which is known as the hierarchy constraint. In other words, annotating a gene to a given class is automatically transferred to all of its ancestors. Third, as the specificity of the functional classes increases, less number of genes are annotated to the more specific functions, this issue is known as the class imbalance problem. Finally, functionally-similar genes may have huge diversity on their measurements through various datasets. To the best of our knowledge, *there is no existing work that handles all of the above challenges in a unified framework.*

To address the above challenges and to improve hierarchical gene function prediction from diverse functional data, we propose **Hi**erarchical **B**ayesian **iN**tegration algorithm (*HiBiN*), a probabilistic framework for integrated analysis of multiple heterogeneous biological data. The major contributions of our new scheme can be summarized as follows:

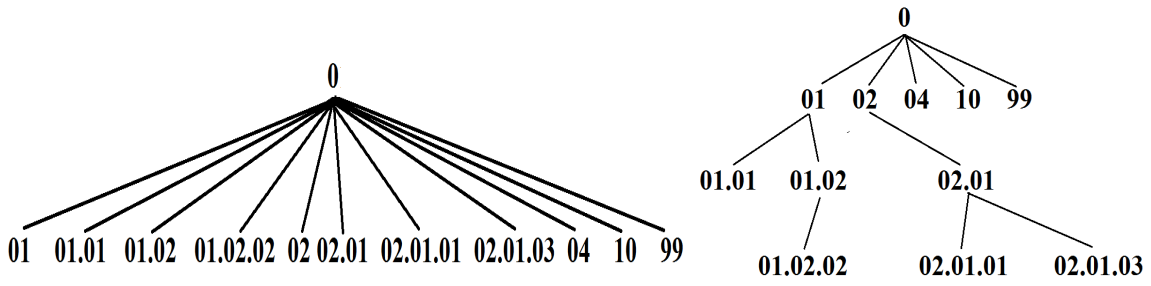
1. To handle the problem of source diversity, the proposed framework integrates multiple heterogeneous data sources to characterize the genes effectively.

2. To predict multi-labels for genes, the HiBiN algorithm allows the prediction of more than one functional class for each gene.

3. To maintain the hierarchy constraint, the parent-child inter-relationships are exploited during the training and the testing phases. More specifically, HiBiN filters out unsuitable genes from percolating to lower levels in the hierarchy.

4. To handle the class imbalance issues, the positive and negative examples for each classifier are chosen based on the hierarchical taxonomy of the classes.

ACM-BCB '11, August 1-3, Chicago, IL, USA



(a) Flat (b) Hierarchical

Figure 1: Flat classification vs hierarchical classification.

2. RELATED WORK

2.1 Single Source Gene Function Prediction

There are two approaches for gene function classifier training: a) Flat approach, where a classifier for each gene function is learned independently (Figures 1(a)) and (b)) Hierarchical approach, where the relationships between gene functions are integrated for gene function classifier training (Figure 1(b)). The flat classification approach may lead to hierarchically inconsistent predictions. On the other hand, the hierarchical approach [1, 2] identifies the relevant positive and negative examples for each class according to the hierarchy. In most of the existing work, a separate binary classifier is learned for each class in the hierarchy.

2.2 Integration of Heterogeneous Data Sources

The value of integrating knowledge obtained from various sources has been illustrated by several studies. Troynskaya et al. [13] proposed MAGIC framework that considers the outputs of several clustering methods applied on each data source separately and incorporates the knowledge of yeast biology experts in the prior probabilities of the Bayesian framework. Marcotte et al. [8] predicted a number of protein functions for *Saccharomyces cerevisiae* based on a heuristic combination of different data types. However, since the confidence levels of protein-protein links are defined on a case-by-case basis, the data sources are combined in a semi-manual and heuristic fashion. Functional linkage networks [3], kernel fusion [7], vector space integration [10] and ensemble systems are other approaches that have been proposed to deal with the data integration problem.

Limitations of the existing methods: the above discussed approaches focus either on discovering interacting sets of proteins only or on integrating multiple sources of data without taking into account the hierarchical relationships between the functional classes. As a consequence, blindly applying these data integration approaches to the hierarchical gene function prediction domain leads to serious inconsistencies due to the violation of the hierarchy constraint governing the functional annotations of genes in GO and FunCat taxonomies. The proposed method overcomes the above limitations.

3. THE PROPOSED FRAMEWORK

Let $\mathcal{G} = \Re^d$ be the d-dimensional input space of genes and $\mathcal{Y} = \{y_1, y_2, ..., y_\mathcal{L}\}$ be the set of $\mathcal{L}$ labels. The hierarchical relationships are defined as follows: Given $y_1, y_2 \in \mathcal{Y}, y_1$ is the ancestor of y_2, denoted by $(\uparrow y_2) = y_1$, if and only if y_1 is a superclass of y_2. We denote by $(\downarrow y_1) = y_2$, the set of children classes of class y_1. The set of labels $\mathcal{L}$ are structured according to a hierarchical structure T (in our case according to a FunCat tree). A gene g is represented with a vector of d different features.

Let a training set $\mathcal{M} = \{< g_1, \mathcal{Y}_1 >, ..., < g_N, \mathcal{Y}_N >\}$, where $g_i \in \mathcal{G}$ is a feature vector for gene i and $\mathcal{Y}_i \subseteq \mathcal{Y}$ is the set of labels associated with g_i, such that $y_i \in \mathcal{Y}_i \Rightarrow y_i' \in \mathcal{Y}_i, \forall(\uparrow y_i') = y_i$. A gene, g, is assigned to one or more functional classes. The assignments are represented through a vector $f_g = (y_1, y_2, ..., y_\mathcal{L}) \in \{0, 1\}^\mathcal{L}$, where $y_i = 1$ if the gene g is annotated with class y_i, while $y_i = 0$ otherwise.

3.1 Boosting Classifiers

ADABOOST.MH [12] is a popular multi-class variant of the AdaBoost algorithm. The algorithm works by transferring a multi-class problem into a binary classification problem by replicating positive instances for the given class labels based on hamming loss [6]. ADABOOST.MH algorithm is presented in detail in [12]. AdaBoost calls a weak learner repeatedly in a series of iterations. At each iteration, s, a weak learner is called to generate a weak hypothesis $\hat{\phi}_s$. In each iteration, a distribution of weights, D, is updated so that the new classifier focuses more on the incorrectly classified instances. After finishing all the iterations, the final hypothesis is generated by summing up all of the weak hypothesis, $\hat{\phi}(g, y) = \sum_{s=1}^{S} \hat{\phi}_s(g, y)$ for $y \in \mathcal{Y} = \{y_1, y_2, ..., y_m\}$.

Adaboost minimizes $E(e^{-c\hat{\phi}(g,y)})$ at $\hat{\phi}(g, y) = \frac{1}{2} log \frac{P(y=1|g)}{P(y=-1|g)}$. Hence,

$$P(g|y = 1) = \frac{e^{\hat{\phi}(g,y)}}{e^{-\hat{\phi}(g,y)} + e^{\hat{\phi}(g,y)}} \quad (1)$$

$$P(g|y = -1) = \frac{e^{-\hat{\phi}(g,y)}}{e^{-\hat{\phi}(g,y)} + e^{\hat{\phi}(g,y)}} \quad (2)$$

Equations 1 and 2 give a number between 0 and 1 that is associated with the likelihood that the gene (g_i) from dataset j is annotated with label k, $p(g_i, \mathcal{M}_j|y_k)$, where $i \in \{1, ..., N\}$, $j \in \{1, ..., Q\}$ and $k = 1, ..., \mathcal{L}$ classes.

3.2 Hierarchical Training and Testing Schemes

The classifier at each class will only be presented with examples that are positive at the parent class of the current class. Hence, the reached examples are positive examples to the current class and/or to the siblings of that class. The training for each classifier is performed by feeding as negative training examples, the positive examples at the parent

of the current that are not positive examples at the current class. It should be noted that the selected negative training examples are the most informative negative examples for training. The testing phase follows the same procedure.

3.3 Bayesian Integration and Classification

For our experiments, the probabilities obtained from boosting classifiers are the likelihood of observing gene i (g_i) associated with dataset j given a specific class. However, the posterior probability is the one of the primary interest:

$$P(y_k|g_i, \mathcal{M}_j) = \frac{P(g_i, \mathcal{M}_j|y_k).P(y_k)}{\sum_{k=1}^{m} P(g_i, \mathcal{M}_j|y_k).P(y_k)} \quad (3)$$

Thus, for each sample, a set of probabilities are obtained which sum to one.

Bayes formula can be used to integrate and compute posterior probabilities for multiple datasets. With the assumption that each dataset is independent but describe the same genes, the integrated likelihood is computed as the product of the individual likelihoods:

$$P(g_i, \mathcal{M}_1, ..., M_Q|y_k) = P(g_i, \mathcal{M}_1|y_k) \times ... \times P(g_i, \mathcal{M}_Q|y_k) \quad (4)$$

Given that the datasets share common genes, the datasets can have some correlation. The probability of a particular gene is given by

$$P(y_k|g_i, \mathcal{M}_1, ..., \mathcal{M}_Q) = \frac{\prod_{j=1}^{Q}[P(g_i, \mathcal{M}_j|y_k)]P(y_k)}{\sum_{k=1}^{m}[\prod_{j=1}^{Q} P(g_i, \mathcal{M}_j|y_k)]P(y_k)} \quad (5)$$

Each non-root class, y_j, has a binary classifier $\hat{\phi}_j$ that is associated with it. The classifier should act as a "filter" to prevent unsuitable examples from spreading out to the lower levels in the hierarchy. Hence, only the test genes that a classifier $\hat{\phi}_j$ decides to belong to y_j are passed to all the binary classifiers corresponding to the children classes of y_j. While the genes that classifier $\hat{\phi}_j$ sees not to belong to y_j are "blocked" and no further analysis is carried out.

HiBiN works by first computing the prior probabilities for all the classes. Next, the children classes of the current class are extracted; then ADABOOST.MH is called on these classes. Then, Bayesian posteriors are calculated for the children classes of the current class.

To make a decision whether a gene is annotated with a particular class or not, we compute the posterior probability for each gene using the following Bayesian decision rule:

$$Decide\ y_k\ if\ P(y_k|g_i)\ >\ P(y_k^{'}|g_i);\ otherwise\ decide\ y_k^{'}$$

where $y_k^{'} = 0$. The computation of $P(y_k^{'}|g_i)$ is similar to the computation of $P(y_k|g_i)$, except that here we compute the probability of the negative class ($y_k^{'}$) and the likelihood given by $P(g|y_k^{'} = 0)$.

4. EXPERIMENTS

4.1 Datasets

We demonstrate the performance of HiBiN for the prediction of gene functions in Saccharomyces cerevisiae. Six different data sources of biomolecular data [14] were integrated. The datasets include (i) gene expression data (Gene-Expr), two types of protein domain data; (ii) protien domain binary data (Pfam-Binary) and (iii) pfam protein domains with log E values computed by the HMMER software toolkit (Pfam-LogE), predicted and experimentally supported protein-protein interaction data from (iv) Von Mering experiments (PPI-STRING), and from (v) BioGrid (PPI-BioGRID) and (vi) pairwise sequence similarity data (Seq-Sim). For the integration purpose, we considered the genes that are common to all datasets. Moreover, for each dataset, we selected FunCat annotated genes for the classes with at least 20 positive examples so that the set of positive examples used for training is not too small. This yielded 1901 yeast genes annotated to 168 FunCat classes distributed across 16 trees and 5 hierarchical levels.

4.2 Baseline Methods

We compared our approach to two baseline algorithms: flat integration and hierarchical single source. In flat integration, the hierarchy constraint is not taken into consideration when building the classifiers, and the integration is performed using the same framework we used for HiBiN method. While the hierarchical single source method is applied on each dataset separately.

To evaluate HiBiN, we used 3-fold cross validation with 100 boosting iterations. Moreover, we tested two strategies to compute the prior probabilities. In one strategy, we computed the prior probabilities without considering the hierarchical structure of the classes. In other words, the prior probabilities are computed by considering the set of annotated genes at each functional label with respect to the total number of genes. This version of the algorithm is called HiBiN$_a$, where a stands for all. In the second variation, the prior probabilities are computed while the hierarchical scheme of the classes is taken into account. In other words, the prior probabilities are computed by considering the set of annotated genes at each functional label with respect to the number of genes that are annotated with the parent functional class of the label of interest. This version of the algorithm is called HiBiN$_p$, where p stands for parent.

4.3 Evaluation Metrics

In order to evaluate our algorithm, we adopted the classical and the hierarchical performance evaluation measures. F_1 measure considers the joint contribution of both precision (P) and recall (R). F_1 measure is defined as follows:

$$F_1 = \frac{2 \times P \times R}{P + R} = \frac{2TP}{2TP + FP + FN} \quad (6)$$

where TP stands for True Positive, TN for True Negative, FP for False Positive and FN for False Negative. When TP=FP=FN=0, we made F_1 measure to equal to 1 as the classifier has correctly classified all the examples as negative examples [5]. Hierarchical measures are defined as follows:

$$hP = \frac{1}{|l(P(x))|} \sum_{p \in l(P(x))} \frac{|C(x) \cap \uparrow p|}{|\uparrow p|} \quad (7)$$

$$hR = \frac{1}{|l(C(x))|} \sum_{c \in l(C(x))} \frac{|\uparrow c \cap P(x)|}{|\uparrow c|} \quad (8)$$

$$hF = \frac{2 \times hP \times hR}{hP + hR} \quad (9)$$

where hP, hR and hF stands for hierarchical precision, hierarchical recall and hierarchical F-measure, respectively.

$P(x)$ is a subgraph formed by the predicted class labels for the instance x while $C(x)$ is a subgraph formed by the true class labels for the instance x. p is one of the predicted class labels and c is one of the true labels for instance x. $l(P(x))$ and $l(C(x)))$ are the set of leaves in graphs $P(x)$ and $C(x)$, respectively. We also computed micro-averaged hierarchical F-measure (hF_1^{μ}) and macro-averaged hierarchical F-measure hF_1^M. hF_1^{μ} is computed by calculating hP and hR for each path in the tree and then applying Equation 9. On the other hand, hF_1^M is computed by calculating hF_1 for each path in the hierarchical structure of the classes independently and then averaging them.

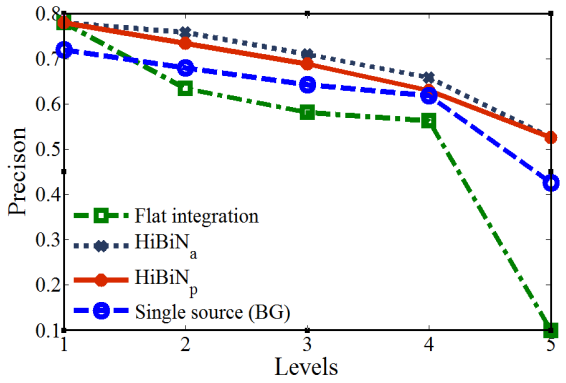


Figure 2: Comparison of the per-level average precision across the five levels of the FunCat taxonomy using flat integration, hierarchical single source (applied on PPI-BG dataset), HiBiN$_a$ and HiBiN$_p$ methods. BG stands for PPI-BioGrid dataset.

The comparison between HiBiN algorithm and the flat method is based on the "per-class" F_1-measure that is obtained by averaging the F_1-measure for all the classes in the FunCat hierarchy for each dataset. In other words, an overall F_1-measure is obtained by computing the F_1-measure for each class separately and then averaging them across all the classes. Furthermore, to get more insights into the performance of the HiBiN algorithm, we performed a level-wise analysis of the precision, recall and F_1-measure on the baselines and the proposed algorithm. In measuring the level-wise performance, level 1 represents the top level in the hierarchy while level 5 is the deepest level in the hierarchy.

4.4 Results

In our experiments, after preprocessing the datasets as described above, the prior probabilities are computed for each label in the FunCat scheme. Prior probabilities are computed based on the number of positive examples annotated with each label obtained from all the datasets. Next, boosting classifiers are used to obtain the likelihoods from the different datasets. Finally, the Bayesian posteriors are computed to obtain the probabilities that are used to make the final decision about the classification.

Table 1 summarizes the results of the comparisons of the average per-class precision, recall and F-measure for the hierarchical single source, flat integration and HiBiN method. We can observe that HiBiN algorithm brings considerable improvement over the baselines. In particular, HiBiN outperforms the baselines in terms of the per-class precision, recall and F-measure. Comparing the flat integration with

Table 1: Average per-class precision, recall and F-measure obtained from hierarchical single source (on each dataset separately), flat integration and HiBiN methods.

Single Source			
Dataset	Precision	Recall	F-measure
PPI-BG	0.5211	0.3176	0.4081
PPI-STRING	0.4882	0.3030	0.3823
Pfam-Binary	0.2977	0.1035	0.3901
Pfam-LogE	0.2244	0.2076	0.3837
Gene-Expr	0.2301	0.2107	0.3805
Seq-Sim	0.2136	0.2074	0.3545
Integration Methods			
Flat	0.3592	0.307	0.4521
HiBiN$_a$	**0.7436**	0.4535	0.6175
HiBiN$_p$	0.7083	**0.4779**	**0.6222**

the hierarchical single source method, there is no clear trend of the winner. For example, the hierarchical single source method performed better on the PPI-BG and PPI-STRING datasets than the flat integration in terms of the per-class precision and recall, while the flat integration performed slightly better than the hierarchical single source method on all the datasets, in terms of per-class F-measure.

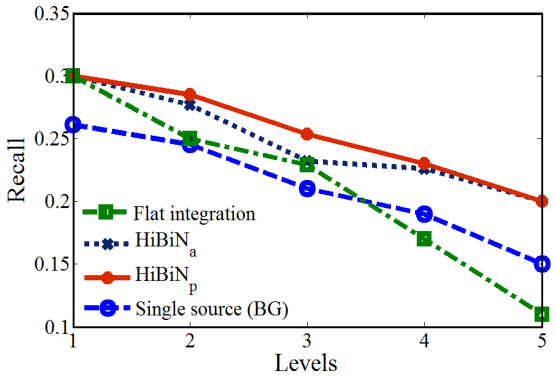


Figure 3: Comparison of the per-level average recall across the five levels of the FunCat taxonomy using flat integration, hierarchical single source (applied on PPI-BG dataset), HiBiN$_a$ and HiBiN$_p$ methods. BG stands for PPI-BioGrid dataset.

In Table 2, we compare the hierarchical precision, hierarchical recall, hierarchical F-micro (hF_1^M) and hierarchical F-macro (hF_1^{μ}) measures for both hierarchical single source and HiBiN methods. Note that hierarchical precision and hierarchical recall are not applicable to flat integration method as flat integration does not take the hierarchy constraint into account. Hence, the obtained predictions may be inconsistent with the hierarchy constraint. As we can observe, HiBiN achieved the best results in terms of all the hierarchical measurements. To a large extent, the two variations of HiBiN have similar performances, which indicates that incorporating the parent-child relationship in the computation of the prior probabilities does not add a noticeable improvement to the overall integration scheme.

A closer look into per-level precision, recall and F-measure

Table 2: Hierarchical precision, hierarchical recall, hierarchical F-micro and hierarchical F-macro measures obtained from hierarchical single source (on each dataset separately) and HiBiN methods.

Single Source				
Dataset	Precision	Recall	F-Micro	F-Macro
PPI-BG	0.9259	0.6071	0.6953	0.7147
PPI-STRING	0.8882	0.5933	0.6846	0.6926
Pfam-Binary	0.8974	0.5912	0.6882	0.7006
Pfam-LogE	0.7340	0.5593	0.6115	0.6348
Gene-Expr	0.7426	0.5801	0.6414	0.6644
Seq-Sim	0.7274	0.5574	0.6067	0.6312
Integration methods				
$HiBiN_a$	**0.9492**	0.6298	**0.7361**	0.7572
$HiBiN_p$	0.9442	**0.6329**	0.7355	**0.7579**

highlights the differences between the proposed method and the baselines. Figures 2, 3 and 4 show the level-by-level performance comparisons in terms of precision, recall and F-measure, respectively, between hierarchical single source applied on PPI-BioGrid dataset, flat integration, $HiBiN_a$ and $HiBiN_p$ methods. Since PPI-BioGrid dataset performed the best for hierarchical single source method, compared with other datasets, we chose it as a representative dataset for the hierarchical single source method. The per-level analysis reveals a degradation in the performance, in all methods, with respect to the depth of the functional classes. However, this degradation is significantly lower when the data integration and the hierarchial relationships among the classes are taken together into account.

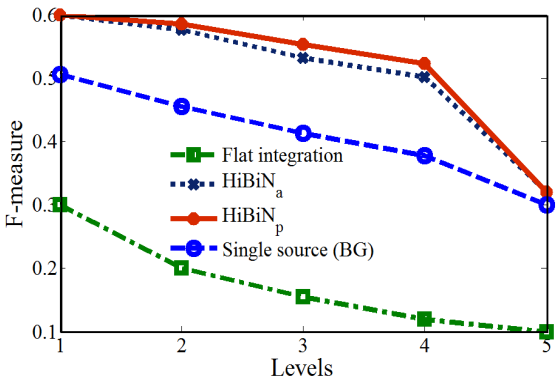


Figure 4: Comparison of the per-level average F-measure across the five levels of the FunCat taxonomy using flat integration, hierarchical single source (applied on PPI-BG dataset), $HiBiN_a$ and $HiBiN_p$ methods. BG stands for PPI-BioGrid dataset.

5. CONCLUSION

In this paper, we developed HiBiN, a general probabilistic framework for gene function prediction through the integration of heterogeneous data sources while maintaining the hierarchy constraint among the functions. Our results showed that the integration can improve the performance of the standard classification-based gene function prediction algorithms. Our future work includes establishing a probabilistic weighting scheme of data sources.

6. REFERENCES

[1] N. Alaydie, C. K. Reddy, and F. Fotouhi. Hierarchical boosting for gene function prediction. In *Proceedings of the 9th International Conference on Computational Systems Bioinformatics(CSB)*, volume 9, pages 14–25, Stanford, CA, USA, August 2010.

[2] Z. Barutcuoglu, R. E. Schapire, and O. G. Troyanskaya. Hierarchical multi-label prediction of gene function. *Bioinformatics*, 22(7):830–836, Jan 2006.

[3] H. N. Chua, W.-K. Sung, and L. Wong. An efficient strategy for extensive integration of diverse biological data for protein function prediction. *Bioinformatics*, 23:3364–3373, December 2007.

[4] T. G. O. Consortium. Gene ontology: tool for the unification of biology. *Nature Genetics*, 25(1):25–29, May 2000.

[5] A. Esuli, T. Fagni, and F. Sebastiani. Boosting multi-label hierarchical text categorization. *Information Retrieval*, 11:287–313, 2008.

[6] G. Jun and J. Ghosh. Multi-class boosting with class hierarchies. In *Multiple Classifier Systems*, pages 32–41, 2009.

[7] G. R. G. Lanckriet, T. D. Bie, N. Cristianini, M. I. Jordan, and W. S. Noble. A statistical framework for genomic data fusion. *Bioinformatics*, 20(16):2626–2635, 2004.

[8] E. M. Marcotte, M. Pellegrini, M. J. Thompson, T. O. Yeates, and D. Eisenberg. A combined algorithm for genome-wide prediction of protein function. *Nature*, 402(6757):83–86, Nov. 1999.

[9] G. Pandey, C. Myers, and V. Kumar. Incorporating functional inter-relationships into protein function prediction algorithms. *BMC Bioinformatics*, 10(1):142+, 2009.

[10] P. Pavlidis, J. Weston, J. Cai, and W. S. Noble. Learning gene functional classifications from multiple data types. *Journal of Computational Biology*, 9(2):401–411, 2002.

[11] A. Ruepp, A. Zollner, D. Maier, K. Albermann, J. Hani, M. Mokrejs, I. Tetko, U. Güldener, G. Mannhaupt, M. Münsterkötter, and H. Mewes. The FunCat, a functional annotation scheme for systematic classification of proteins from whole genomes. *Nucleic Acids Research*, 32(18):5539–5545, Oct 2004.

[12] R. E. Schapire and Y. Singer. BOOSTEXTER: Aboosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168, 2000.

[13] O. G. Troyanskaya, K. Dolinski, A. B. Owen, R. B. Altman, and D. Botstein. A Bayesian framework for combining heterogeneous data sources for gene function prediction (in Saccharomyces cerevisiae). *Proceedings of the National Academy of Sciences of the United States of America*, 100(14):8348–8353, July 2003.

[14] G. Valentini. True path rule hierarchical ensembles for genome-wide gene function prediction. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 8:832–847, 2011.