Predicting gene functions from multiple biological sources using novel ensemble methods

Chandan K. Reddy* and Mohammad S. Aziz

Department of Computer Science,
Wayne State University,
Detroit, MI, 48084, USA
Email: reddy@cs.wayne.edu
Email: maziz@wayne.edu
*Corresponding author

Abstract: The functional classification of genes plays a vital role in molecular biology. Detecting previously unknown role of genes and their products in physiological and pathological processes is an important and challenging problem. In this work, information from several biological sources such as comparative genome sequences, gene expression and protein interactions are combined to obtain robust results on predicting gene functions. The information in such heterogeneous sources is often incomplete and hence making the maximum use of all the available information is a challenging problem. We propose an algorithm that improves the performance of prediction of different models built on individual sources. We also develop a heterogeneous boosting framework that uses all the available information even if some sources do not provide any information about some of the genes. We demonstrate the superior performance of the proposed methods in terms of accuracy and F-measure compared to several imputation and integration schemes.

Keywords: gene function prediction; data integration; ensemble methods; boosting.

Reference to this paper should be made as follows: Reddy, C.K. and Aziz, M.S. (2015) ‘Predicting gene functions from multiple biological sources using novel ensemble methods’, *Int. J. Data Mining and Bioinformatics*, Vol. 12, No. 2, pp.184–206.

Biographical notes: Chandan K. Reddy is an Associate Professor in the Department of Computer Science at Wayne State University. He received his PhD from Cornell University and MS from Michigan State University. His primary research interests are data mining and machine learning with applications to healthcare analytics, bioinformatics and social network analysis. His research is funded by NSF, NIH, DOT, and the Susan Komen for the Cure Foundation. He received the Best Application Paper Award at SIGKDD conference in 2010, and was finalist of the INFORMS Franz Edelman Award Competition in 2011. He is a senior member of IEEE and member of ACM.

Mohammad S. Aziz completed his Masters in Computer Science from Wayne State University. His research interests include data mining and machine learning with applications to bioinformatics.

This paper is a revised and expanded version of a paper entitled 'Robust prediction from multiple heterogeneous data sources with partial information' presented at the '18th ACM Conference on Information and Knowledge Management (CIKM)', Toronto, Canada, October 2010.

1 Introduction

In the field of functional genomics, the functional classification of genes that are unannotated and improving the existing gene functional annotation catalogs is an important and challenging problem. Due to its ability to detect previously unknown role of genes and their products in physiological and pathological processes, functional classification plays a vital role in molecular biology (Re and Valentini, 2010). Nevertheless, the development of an automatic model for such a classification task is still limited by the intrinsic difficulty of the task and lack of a reliable mechanism that can effectively leverage partial sources of information. However, researchers from different domains are extracting some task specific information for the same model organism and its genes. This lead to the availability of different types of biomolecular data, ranging from expression profiles to phylogenetic gene-specific evolution rates and many others. Such vast amounts of data can, in principle, provide useful information for the automated assessment of the functional role of genes. The extent to which the presence of a specific type of experimental data could improve the classification performance significantly varies for specific gene and the particular biomolecular process under investigation. Though the availability of multiple sources have a tremendous potential for improving the performance of functional classification, it also poses some challenges which include heterogeneous non-compatible features and unavailability of information for all the genes in different datasets. In this paper, *we primarily focus on improving the performance on the functional classification by utilising multiple sources of information about a set of genes*.

Recently, a systematic evaluation of classification performance using different combination rules suitable to merge the output of gene function classifiers trained on different data sources is presented (Re and Valentini, 2010). Due to the heterogeneous nature of different data sources, it is possible that different types of classification models may perform better on different datasets. In this work, we build ensemble models to combine the information from different sources and evaluate the performance of such ensemble models. The intuition behind combining the predictions of different classifier comes from the fact that different base classifier may perform better for different samples. Combining the final decisions of a number of classifiers can obtain a strong classification model that (on an average) performs well for most of the genes.

The availability of multiple sources of information have a tremendous potential for improving the performance of functional classification. Though the integration approaches discussed above demonstrate good potential to produce a robust model, they suffer from one limitation. They can be only applied to the common genes that are available across all the sources. One of the key challenges in this domain is that many sources contain only partial information and one can rarely see all the information available in a given source. Some of the sources potentially consist of information about certain genes that is not available in some of the other sources. Using only the common information implies that we are not using all the available information which might be vital for building a better prediction model. Furthermore, when the prediction model is used on a particular test case, it is unlikely that the testing gene will have information for all the sources. In such cases, the prediction model generated from a model built on common set of genes will be suboptimal. Thus, the need for a general framework which can use all the available information and also capable of making prediction for the test case for which not all the information is available is required. This motivated us to build a generalised framework for the integrated predictive modelling in the presence of uncommon data objects. Though some kernel-based methods try to handle this problem via imputation at the early stage, a systematic and generalised framework which exploits the partial information that is available in multiple sources in a better manner is not available in the literature. This is precisely the main objective of the work that is being proposed in this paper. Generally, in such cases, only the common samples are taken for the study thereby leading to a significant loss of potential information during integration. There are some works that impute the data to make it complete and then using the complete datasets for further integration process (Williams and Carin, 2005). These works typically make a few reasonable assumptions and integrate the information analytically calculating the kernel matrix from incomplete data. In case of very few missing elements, semi-definite programming can be used to complete the kernel matrix (Graepel, 2002). In this paper, we develop a new approach to solve this problem of integration from multiple sources in the presence of only a portion of samples that are common across all the sources. In other words, majority of the samples are available only in fewer sources. Here, we refer to a data point as 'uncommon' when it is not available in all the sources.

The rest of the paper is organised as follows: Section 2 describes some of the related works related to our problem. Section 3 describes proposed algorithms for dealing with a set of common genes and with partial information. Section 4 discusses the experimental setup and explains the sources from which different datasets were extracted. The results of the functional classification task are given in Section 5 and finally, Section 6 concludes our discussion.

2 Related background

2.1 Notations used

Table 1 gives the notations used in this paper.

2.2 Types of data integration

Three basic approaches for data integration for the task of class prediction have been proposed in the literature:

1 *Early integration*: Methods such as Vector Space Integration (VSI), which basically integrate all the data sources into a single data file and predict classes with all the features available. Early integration produces a single, potentially more informative dataset which can then be used for the prediction task (des Jardins et al., 1997).

Other such methods were based on modelling networks of functional relationships between proteins where graphical models provide a probabilistic framework for data integration (Karaoz et al., 2004).

2 *Intermediate integration*: which is typically based on Kernel Fusion (KF) methods. The integration is done during the training phase itself. Individual kernels on different sources are learned first and the final classifier is built on the composite kernel that is built after combining the individual ones (Lanckriet et al., 2004). To exploit the heterogeneity of the data, weighted functional linkage graph is generated using different sources of information (Zhao et al., 2008).

3 *Late integration*: This typically models individual sources and then combines the knowledge from these individual models and builds a final classifier (Kuncheva et al., 2001; Roli et al., 2001). Methods such as decision templates, different types of weighted majority voting using linear or logarithmic weight combination and ensemble methods like bagging, boosting and random forests fall into this category (Dietterich, 2000; Polikar, 2006; Krishnaraj and Reddy, 2008). The methods that are being proposed in this paper fall into this category.

Table 1 Notations used in this paper

Notation	*Description*
N	Number of datasets
m	Number of total data points
D_i	i-th dataset
m_i	Number of data points in D_i
d_k	k-th data point
w_{ki}	Weight of the data point d_k for D_i
C_{ij}	Weak classifier at j-th iteration for D_i
M_i	Strong classifier generated from D_i
λ	Indicator matrix
λ_{ki}	Indicator variable for data d_k in D_i
λ_i	Indicator variable for test data in D_i
c_{ij}	Weight of the weak classifier C_{ij}

2.3 Ensemble methods for integration

Recently, ensembles of classifiers have been gaining a lot of interest because of their excellent generalisation performance. The intuition for using the ensemble technique is that, if the base classifiers composing the ensemble are diverse, then they are expected to make different errors and hence, the ensemble output produced by these classifiers is expected to reduce the error through some form of weighted averaging (Kuncheva and Whitaker, 2003).

Weighted majority voting (Kittler et al., 1998): Weighted majority voting is the most widely used late integration method because of its inherent simplicity, natural mapping to the problem and superior generalisation performance. There are many general and problem specific majority voting schemes that can be used for the problem that are addressed in this paper. Let, a trained classifier from *i*-th dataset and *j*-th type (any base classifier) computes a function d_{ij}^{k} for a specific class φ_k such that $d_{ij}^{k}: X \rightarrow [0,1]$. An ensemble combines the outputs of *T* base learners using a suitable combining function *g* to compute the posterior probability μ_k for a given class φ_k.

$$\mu_k(x) = g(d_{11}^{k}(x), d_{12}^{k}(x), \ldots, d_{Nc}^{k}(x)) \tag{1}$$

where *N* is the number of information sources and *c* is the number of the base classifiers. In this paper, we made the ensemble decision using the different schemes described below. Please note that these methods are being used as baseline methods for comparison purposes.

Ensemble linear: In Ensemble Linear, the combined result is generated using a weighted combination of the decisions of base classifiers, where the weight of the base classifier in the consensus is calculated linearly using the following equation:

$$\mu_k(x) = \sum_t \omega_t d_{it}^{k}(x), \; where \;\; \omega_t = \frac{C_t}{\sum_t C_t} \tag{2}$$

where C_t is the accuracy of the *t*-th base classifier, when optimising for accuracy (or F-measure).

Ensemble logarithmic: Ensemble Logarithm is similar to Ensemble Linear except that the weights of the base classifiers are calculated logarithmically as follows:

$$\omega_t = \frac{p_t - ln\,\varepsilon}{\sum_t p_t - ln\,\varepsilon}, \; where \;\; p_t = ln\frac{C_t + \varepsilon}{1 - C_t + \varepsilon} \tag{3}$$

where ε is a small constant value which avoids the indeterminate form.

Ensemble of similar classifiers: Ensemble of similar classifiers is a popular method. In this case, the combined decision is produced using the same type of classifiers. i.e. SVM ensemble is generated by *m* base (SVM) classifiers trained on *m* different datasets. The consensus decision is obtained as follows:

$$\mu_{ki}(x) = \sum_t \omega_t d_{it}^{k}(x) \tag{4}$$

where $\mu_{ki}(x)$ is the *i*-th Ensemble of similar classifier for the class φ_k. For example, it is composed of the *i*-th model from each dataset.

Ensemble of the best classifier: The best classification model on different datasets can be ensembled to produce a better result compared to the ensemble of similar classifiers. The consensus decision in this case is given as follows:

$$\mu_k(x) = \sum_t \omega_t^{'} d_{ts}^{k}(x), \; where \;\; s = argmax_j \omega_j \tag{5}$$

where ω's are calculated for the base classifier models for a particular dataset as explained earlier. However, in the end, *N* such classifiers will be chosen when the previous normalisation will not hold anymore. ω's are the weights normalised across the *N* best classifiers from these datasets.

Decision templates (Kuncheva et al., 2001): Decision template is an approach which makes use of all the base classifiers trained on each of the *m* datasets. The decision profile *DP*(*x*) for an instance *x* is a matrix composed by the $d_{t,j} \in [0,1]$ elements representing the support given by the *t*-th classifier to class φ_j. Decision templates DT_j are the averaged decision profiles obtained from X_j, the set of training instances belonging to the class φ_j:

$$DT_j = \frac{1}{|X_j|} \sum_{x \in X_j} DP(x) \tag{6}$$

The similarity *S* between the decision template DT_j for a class φ_j and the decision profile for a given test instance *x* is:

$$S_j(x) = 1 - \frac{1}{T \times C} \sum_{t=1}^{T} \sum_{k=1}^{C} \left[D_j(t,k) - d_{t,k}(x) \right]^2 \tag{7}$$

and the final decision of the ensemble is calculated by assigning the test instance to the class with the largest similarity:

$$D(x) = argmax_j S_j(x) \tag{8}$$

2.4 Data imputation for partial data

In the literature, there were only a few attempts that were made on incorporating information in the presence of several uncommon data points. Most of the work on handling such scenarios is available in the kernel fusion methods where the kernel matrix is integrated to combine the information from multiple sources. Most of these methods treat this as a missing data problem to calculate the missing features with the help of observed features to subsequently compute the kernel matrix. There are few propositions that fill the missing entries in the kernel matrix directly. In our study, we comprehensively compared the proposed method with the following kernel fusion alternatives:

1 *Unconditional mean imputation (UMI):* For a data point that is not in a particular source, the feature values are imputed with the average of the feature values of the points that are present in that source. After getting all the feature values, we aggregate the kernel matrix to get a single kernel matrix and use SVM classifier on it to make the prediction.

$$f_{d_k}^j = \frac{1}{|S|} \sum_{d \in S} f_d^j \tag{9}$$

where *S* is the set of data that has the value for feature *j*-th feature.

2 *Weighted summation imputation (WSI):* The feature values for the data are not imputed in the source. Rather, for a data point that is missing in a source, the kernel matrix entries of that point for this kernel are imputed as a weighted combination of the average of the entries of that particular kernel matrix and the average of the entries for that samples in other kernel matrices where the gene is present. In our experiment, we used 50% weight for both these values.

3 *Nearest neighbour imputation (NNI):* In this method, the kernel matrix entries are directly imputed rather than imputing the feature values. First, the kernel matrices for the different sources are generated from the genes present in those sources. At this juncture, for a gene that is not present in a particular source does not have any kernel matrix entry. Using the other sources where the feature values are present for that gene, the nearest neighbour is obtained. Then, in the source where that gene was missing, the kernel matrix entry of the nearest neighbour is replicated. However, there was no entry of the nearest neighbour for the new gene which is filled by the current highest entry of the matrix that originally corresponds to the most similar genes in the source.

3 Proposed methods for data integration

We will now describe the details of the two algorithms that are proposed to handle the integration of multiple heterogeneous datasets. The first algorithm, 'DecBoost', is developed for the case where complete information is available. In other words, the information about all the genes are available in all the data sources. For the partial data case, we develop 'HeteroBoost' which can handle the integration in the presence of partial information. In this case, some genes are available in only few data sources and hence will not have information from the rest of the data sources. When all the data is integrated, the matrix will become incomplete.

3.1 Decision boosting

Typically, different classifiers work well on different sources and their corresponding classification models show superior performance compared to others (Caruana and Niculescu-Mizil, 2004). In this objective, we will build boosting-based integration framework for solving the prediction problem in the presence of multiple heterogeneous data sources. 'Heterogeneous' refers to different types of data stored in different sources such as categorical, numerical, string, network data etc. Appropriate features along with their similarity measures relevant to the prediction task at hand will be decided before the integration step and the data is stored in a matrix format. Our approach falls into the category of late integration where we combine the decisions of different classifiers (Aziz and Reddy, 2010). After building individual models for each data source, we take the decisions made on individual genes and use them as features for the next stage. We will then apply boosting method using the new feature representation which is comprised of these individual classifiers. *Even if strong classifiers are used to build local models for individual sources, they are considered to be weak learners for the final classification model that is being built in the latent integration space.* This is because, most of the times these sources are considered to be weak sources of information for the final prediction task.

Such a two-level integration scheme can provide optimal results since the first level (local modelling) can produce reliable models for individual sources and the second level can produce the most optimal combination of such models. Since there is no single classifier model that is optimal for all the datasets, we will use many different classifiers for individual sources and create individual features from the decisions (or probabilities) of each of the classifiers. In the second level, one can then use a standard classification algorithm (boosting in our case) with this new feature representation. Though some of the steps in this process appear to be available in the literature, such an approach has not been rigorously investigated in the context of integrating information from multiple heterogeneous sources.

The effectiveness of an ensemble method depends on the diversity of the base classifiers being used. In other words, if different base classifiers make error on different genes, then the combined result can get rid of some errors and improve the final accuracy of the ensemble models. To ensure the improved accuracy when combining models from multiple sources, we propose to boost the decisions from these individual sources using AdaBoost. The AdaBoost algorithm (Freund and Schapire, 1996) is an efficient, simple and easy to manipulate additive modelling technique that can potentially use any available weak learner. Boosting algorithms combine weak learning models that are slightly better than random models. It is an ensemble method that generates multiple classifiers from a base learner and ensembles them for building the best classifier (Freund and Schapire, 1997). The basic idea of boosting is to repeatedly apply the weak learner to modified versions of the data, thereby producing a sequence of weak classifiers for $t = 1,2,\dots,T$, where T denotes predefined number of iterations. Each boosting iteration performs the following steps: (a) Fits a weak learner to the weighted training set and (b) Computes the error and updates the weights for every data point. The final model obtained by boosting algorithm is a linear combination of several weak learning models.

For a particular gene, we will consider all the decisions from the base classifier models as different feature values and construct a vector of these decisions. We improve the quality of these decisions by applying a boosting algorithm on such a decision matrix. We thus call the proposed algorithm as '*DecBoost*'. In simple terms, we boost the decisions made from individual data sources and provide a much robust prediction result. The details of the DecBoost are shown in Algorithm 1. It is a modification of AdaBoost where the weak learners are the different types of classifiers that are run on various sources. It should be noted that this method is effective only when each source contains enough information to make a decision about a given gene.

3.2 Heterogeneous boosting for integration with partial information

In practice, one can not obtain all the required information about a particular gene from all the sources. Often, some sources deliver information about some of the genes whose information is not available from other sources. In other words, when all the sources are combined, there will be some missing information about certain genes with respect to some sources. Most of the current research work in data integration primarily focuses on integrating information when all the sources contain the information about a gene. That is, for the sake of convenience, researchers primarily deal with common genes that are available in all the data sources. In certain cases, utilising only the common information will potentially produce inferior prediction result when there are several genes with partial information. Some work on handling partial information is available in the kernel methods literature (Sharpe and Solly, 1995; Smola et al., 2005; Chechik et al., 2008) where the kernel matrix is integrated to combine the information from multiple sources. Most of these methods treat it as a missing data problem to calculate the missing features with the help of observed features to subsequently compute the kernel matrix (Graepel, 2002; Williams and Carin, 2005). Though some kernel-based methods can handle this problem via imputation at the early stage, they only work well when there is less amount of incomplete data (Little and Rubin, 2002). However, in practice, when several sources are being integrated, the amount of information about a particular gene vastly differs and we will find significant amount of missing information. In such cases, imputation methods do not work well during integration and will yield suboptimal solutions. Most importantly, when the prediction model is used on a particular test case, it is unlikely that the test case will have information for all the sources. Hence, our idea is to rely only on particular sources available for those uncommon genes and thus, give more importance (or weight) to that source during the training phase for that particular gene.

Algorithm 1: DecBoost

```
1: Input: Datasets $D_1 \ldots, D_N$ with the common data points $x_1 \ldots, x_N$
2: Output: Final strong classifier $h(x)$
3: Procedure:
4: From each dataset $D_i$, generate $M$ weak classifiers $C_{i,1} \ldots, C_{i,M}$. [Generating
the classifier pool]
5: Initialise weight $w_{1,i} = \frac{1}{n}$ [Boosting starts here]
6: For $t = 1..T$:
      (a) Find the best classifier $C_t$ from the pool using the weighted error
   criterion $\varepsilon_t = \sum_i w_{t,i} |C_t(x_i) \neq y_i|$
      (b) Update the weights $w_{t+1,i} = w_{t,i} * exp[c_t . |C_t(x_i) \neq y_i|]$ where $c_t = log \frac{(1-\varepsilon_t)}{\varepsilon_t}$
      (c) Normalise $w_{t+1,i} = \frac{w_{t+1,i}}{\sum_i w_{t+1,i}}$
7: Return classifier $h(x) = \sum_t c_t . C_t(x)$
```

We propose a heterogeneous boosting-based integration framework that will exploit all the available (including partial) information from multiple data sources. To achieve this goal, we propose a novel objective criterion which will emphasise the importance of genes with partial information compared to the common ones. We will also modify the re-weighting scheme in the following manner: if a gene is present in only one source out of c sources, the importance of it will be increased by *c* times while modelling that gene. We plan to use weighted classifiers and incorporate the penalty term that takes into account the information about the importance of each gene. The increase of weight of the misclassified gene will be inversely proportional to the number of datasets that contain the information about the gene. *The basic intuition here is that the algorithm will give more importance to a gene if it is available only in one source compared to one being available in many sources*. At the end of each boosting iteration, the instances are re-weighted in such a way that the misclassified objects get a higher weight so that the next weak classifier gives more importance to those objects that were misclassified in the previous iteration.

In boosting algorithm, a strong classifier is built as a combination of a number of weak classifiers, where each classifier is chosen at every iteration if its accuracy is greater than 50%. At the end of each iteration, the samples are re-weighted in such a way that the misclassified samples get a higher weight so that the next weak classifier shows a better performance on those samples that were misclassified in the previous iteration. We modified the boosting algorithm to build a strong model out of a dataset using all its data points in such a way that increment in all the misclassified samples is not equal. We use the term ‘weak classifier’ to refer to the classifier chosen in each iteration during the boosting process of a particular dataset, the term ‘strong classifier’ to refer to the classifier that is generated out of a Heterogeneous dataset and the term ‘stronger classifier’ to refer to the final classifier that is a weighted combination of the strong classifiers.

For getting the stronger classifier from dataset D_i, we modify the AdaBoost weight updation in such a way that it holds the following two properties:

- The weight for the misclassified examples is increased.
- The increase of the weight of the misclassified data point will be inversely proportional to the number of datasets that contain the information about the data point.

The basic intuition behind the second criteria is that the algorithm does want to give more importance to a data point when it is available only in one source compared to data point that is available in many sources. Note that, if it is misclassified in other datasets during an iteration, then its weight is increased in that dataset as well. Thus, it is unlikely that a data point is neglected in all the strong models thus making the stronger model more general and diverse.

Let λ denote an indicator matrix such that λ_{ki} is 1 if the gene d_k is in dataset D_i and 0 otherwise. We define $\overline{\lambda_i}$ as the average number of datasets in which the data from D_i is present and is calculated as follows:

$$\overline{\lambda_i} = \frac{\sum_k (\lambda_{ki} * \sum_{i'} \lambda_{ki'})}{\sum_k \lambda_{ki}} \tag{10}$$

We modify the re-weighting scheme in AdaBoost to follow the two above mentioned criteria as follows:

$$\frac{1-\varepsilon_j}{\varepsilon_j} = 1 + \frac{(1-2*\varepsilon_j)}{\varepsilon_j} \tag{11}$$

where ε_j is the error rate for that iteration. Hence, the increment amount is $\frac{(1-2*\varepsilon_j)}{\varepsilon_j}$ which is positive since $\varepsilon_j < 0.5$. We varied this increment amount based on the number of datasets that the gene is present in:

$$\frac{(1-2*\varepsilon_j)}{\varepsilon_j} * \frac{\overline{\lambda_i}}{\sum_{i'} \lambda_{ki'}} \tag{12}$$

It should be noted that the multiplying factor is defined in such a way that the aforementioned two criteria are fulfilled and it retains the property of balancing i.e. the product of all the multiplication factors for a dataset is 1. Let mf_k denotes the multiplying factor of k-th data point, then

$$\prod_k mf_k = \prod_k \frac{\overline{\lambda_i}}{\sum_{i'} \lambda_{ki'}} = \prod_k \frac{\sum_k (\lambda_{ki} * \sum_{i'} \lambda_{ki'})}{\sum_k \lambda_{ki}} * \frac{1}{\sum_{i'} \lambda_{ki'}} = 1 \tag{13}$$

Algorithm 2: HETEROBOOST

```
1: Input: Datasets $D_1 \ldots, D_N$, samples $d_1 \ldots, d_m$ and the indicator matrix $\lambda$
2: Output: Final strong classifier $M$
3: Procedure:
4: for $i = 1..N$:
5:     Initialise weight $w_{ki} = \frac{1}{m_i}$
6:     for $j = 1..T$:
7:       (a) $C_{ij} \leftarrow argmin_{C_{ij} \in H} \sum_k \lambda_{ki} w_{kj} |C_{ij}(d_k) \neq y_k|$
8:       (b) for $k = 1..m$:
9:         (i) $c_{ij} = log(1 + \frac{(1 - 2 * \varepsilon_j)}{\varepsilon_j} * \frac{\overline{\lambda_i}}{\sum_{i'} \lambda_{ki'}})$
10:        (ii) $w_{ki} = w_{ki} * exp\left[c_{ij} . |C_{ij}(d_k) \neq y_k|\right]$
11:      (c) Normalise weights: $\forall_k \; w_{ki} = \frac{w_{ki} \lambda_{ki}}{\sum_k w_{ki} \lambda_{ki}}$
12:    end for
13:    strong classifier $M_i(x) = \sum c_{ij} . C_{ij}(x)$
14: end for
15: return stronger classifier $M(d) = \frac{\sum_{i=1}^n F_i M_i(d) \lambda_i}{\sum_{i=1}^n F_i \lambda_i}$
```

During the test phase, our method will consider only those sources where the information about the test cases is available and then obtain a weighted ensemble model out of those sources. In other words, there will be no imputation performed in the sources that do not have information about that particular gene. The final outcome is calculated as follows: $M(d) = \frac{\sum_{i=1}^n F_i M_i(d) \lambda_i}{\sum_{i=1}^n F_i \lambda_i}$ where F_i is the evaluation metric (such as accuracy or F-measure) that is being measured for the strong boosted classifier M_i and d is the test gene.

3.2.1 Advantages of the proposed framework

For integrating multiple sources, there had been a lot of research works in developing new kernel-based methods. Our approach has the following advantages compared to the kernel-based fusion methods.

1 *Modularity:* It can efficiently handle dynamic sources because of our modular approach. It is flexible in terms of adding/removing/updating data sources. We will perform the local modelling only on the modified data source without modifying the features obtained from any other source. Only the second level classifier built will have to be retrained. This is a big advantage over the kernel-based fusion techniques (Varma and Babu, 2009) where the system has to be retrained completely when one particular source is modified since the kernel matrix has to be recomputed again. This a tedious task in many practical applications where the data sources change constantly. One can also potentially use hierarchical models in combination of these ensemble methods (Reddy and Park, 2011; Alaydie et al., 2010).

2 *Scalability:* Kernel methods obtain the composite kernel by optimising using semi-definite programming which is a computationally expensive and hence, it is not a viable solution for large scale problems. Though there had been some works on efficiently optimising kernels for large-scale problems (Sonnenburg et al., 2006; Zien and Ong, 2007), domain experts hesitate to use such complicated methods when simpler and more interpretable methods can be used. Hence, there is a huge gap between making the theoretical research being made available to the practitioners. Our approach can conveniently build upon some of the existing tools that the practitioners are already using and achieve the final goal in faster time and is easily parallelisable (Palit and Reddy, 2012).

3 *Robustness to partial information:* For handling partial information, the imputation-based methods follow an indirect two step approach by first imputing the values and then building models thus propagating any errors accumulated during the imputation phase to the next phase. Our approach is a direct method where we build models without imputation and build models directly from the data available to us.

4 *Interpretability:* Quite often, the domain experts would like to know the impact of some of the data sources for the performance of integrated model.

4 Experimental setup

In this section, we will first describe the various datasets that were used in our experiments. We will also discuss the experimental setup for evaluating the proposed framework.

4.1 Datasets

In order to evaluate the effectiveness of the proposed integration framework, we used the genes from *S. cerevisiae* (yeast) for which information is available from different sources. We have chosen *S. cerevisiae* because it is the most widely studied model organism for which vast amounts of bio-molecular data are available (Rhee et al., 2008). We have used the following six bio-molecular datasets to evaluate the proposed gene function prediction methodology:

1 *BioGRID:* The source of this protein–protein interaction data is BioGRID database (Stark et al., 2006) that collects PPI data from both high-throughput studies and conventional focused studies. BioGRID houses high-throughput two-hybrid, mass spectrometric protein interaction data and synthetic lethal genetic interactions obtained from synthetic genetic array and molecular barcode methods, as well as a vast collection of thoroughly validated physical and genetic interactions from the literature. It is binary data which represents the presence or absence of protein–protein interactions.

2 *Sequence:* The pair-wise similarity data from the Smith-Waterman algorithm represents homological functional relations that exist between genes belonging to the same functional classes. Each data value computed from the Smith-Waterman log-E values between a pair of yeast sequences that express the pair-wise similarities between the genes.

3 *DomainBinary:* DomainBinary consists of protein family data (Finn et al., 2008). The basic idea of using such information is the tight connection between the protein structure and its ability to perform a particular biological task. Proteins comprise of structured regions usually referred to as domains joined by unstructured regions named loops which can be a potential source of information about the functional role of a gene. Each specific domain constituting a protein is reposed to the realisation of a specific task (either structural or biochemical) and thus the presence of particular kind of domain in the protein structure could be of vital importance in the prediction of its function. For each gene product, the presence or absence of 4950 protein domains obtained from the DomainBinary database is stored as a binary vector.

4 *DomainLogE:* DomainLogE is also processed from the previous data source. The E-value assigned to each gene product is computed from a collection of profile-HMMs, each of which is trained on a specific domain family. The E-values are obtained from the HMMR software toolkit (can be obtained from http://hmmer.janelia.org).

5 *Gasch:* Gasch gene expression dataset is generated by merging the experiments of Spellman (gene expression measures relative to 77 conditions) (Spellman et al., 1998) with the transcriptional responses of yeast to environmental stress (173 conditions) by Gasch (Gasch et al., 2000).

6 *STRING:* This is also a protein–protein interaction data that is collected from a different source named STRING (von Mering et al., 2003). It contains binary PPI data from yeast two-hybrid assay, mass-spectrometry of purified complexes, correlated mRNA expression and genetic interactions.

From Tables 2 and 3, we can observe that there are many uncommon genes along with only 1901 common genes (genes that are present in all the six data sources). In Table 3, the column labelled 'Total' indicates the number of genes that are available in only few datasets. In other words, the first row in Table 3 indicates the total number of genes that are available in one and only one dataset (105) and which dataset they are present (BioGRID-66 and Gasch-39). Table 3 provides a complete distribution of the information available about the genes across all the datasets. One can also see information about 636 genes is available in only two datasets (632 from BioGRID, 634 from Gasch and six from STRING).

Table 2 Six bio-molecular datasets used in our experiments

Data source	*No. of samples*	*No. of features*
BioGRID	4531	5367
Sequence	3527	6349
DomainBinary	3529	5724
DomainLogE	3529	4950
Gash	4524	250
STRING	2338	2559

Table 3 The distribution of genes across the six bio-molecular datasets used in this paper

In dataset	*Total*	*BioGRID*	*Sequence*	*DomainBinary*	*DomainLogE*	*Gasch*	*STRING*
1	105	66	0	0	0	39	0
2	636	632	0	0	0	634	6
3	401	395	5	6	6	397	394
4	118	57	118	118	118	61	0
5	1504	1480	1503	1504	1504	1492	37
6	1901	1901	1901	1901	1901	1901	1901
Total	4665	4531	3527	3529	3529	4524	2338

4.2 Functional annotations

We used functional annotations collected from the Functional Catalogue (FunCat) database (Ruepp et al., 2004) to associate each of the genes in the six aforementioned datasets to a functional class. We have chosen FunCat since it consists of annotations primarily based on experimental evidence, which allows us to minimise the impact of non-experimental functional annotations (Re and Valentini, 2010). The Functional Catalogue comprises of hierarchically structured controlled vocabulary of functional categories in a forest like structure and was originally developed to describe yeast functional processes. In order to reduce the number of candidate classes and to quantitatively compare the results of other methods proposed in the literature, we considered the first level (that is the most general and wide functional classes of the overall taxonomy) of classes that are represented by at least 20 genes. The 15 classes chosen for our experiments are shown in Table 4. The numbers that preceded the functional classes are the numbers from the FunCat annotations.

Table 4 Functional classes that contain significant number of genes

FunCat No.	*Functionality*
1	METABOLISM
2	ENERGY
10	CELL CYCLE & DNA PROCESSING
11	TRANSCRIPTION
12	PROTEIN SYNTHESIS
14	PROTEIN FATE (folding, modification)
16	PROTEIN WITH BINDING FUNCTION OR COFACTOR REQUIREMENT
18	REGULATION OF METABOLISM & PROTEIN FUNCTION
20	CELLULAR TRANSPORT & TRANSPORT ROUTES
30	CELLULAR COMMUNICATION/SIGNAL TRANSDUCTION MECHANISM
32	CELL RESCUE, DEFENSE & VIRULENCE
34	INTERACTION WITH THE ENVIRONMENT
40	CELL FATE
42	BIOGENESYS OF CELLULAR COMPONENTS
43	CELL TYPE DIFFERENTIATION

5 Results and discussion

First, we identified 1901 genes that are common across all the datasets and the rest of the 2764 other genes that are present in only fewer datasets (see Table 3). We performed threefold cross-validation for reporting our results on the test data. We randomly divided both the common and uncommon genes into three folds. The combination of two folds is used for training and the rest for testing. To get the combined result, one fold of the common genes is merged with one fold of the uncommon genes to make one fold of the combined gene set. Because of the class imbalance problem, the models often produce poor result for F-measure for the target class. To tackle the class imbalance issue, we pre-processed the training data before the training process to undersample the majority classes and oversample the minority class using Synthetic Minority Oversampling TEchnique (SMOTE) (Chawla et al., 2002).

While working with the kernel fusion methods, we used the same fold generated for the heterogeneous boosting. For UMI, we first imputed the missing feature values and then generated complete kernel matrices for different datasets. For the other two methods, we imputed the kernel matrix to obtain a complete kernel matrix. In either case, we have a set of full kernel matrices. To get the integrated kernel matrix we computed: (a) simple summation of those kernel matrices and (b) weighted summation of those kernel matrices based on the individual accuracy in corresponding datasets. We performed two separate sets of experiments for the common genes and the uncommon genes. Here, we discuss the results of these experiments separately.

5.1 Integration results with common genes

While choosing the classification methods as base learners, we used four state-of-the-art machine learning methods: SVM, Naive Bayes, Decision Tree and Adaboost (with decision stumps). We identified 1901 genes that are common across all the six sources. While choosing the base learners, we used threefold cross-validation where each fold was generated using stratified random sampling. In this manner, for each of the 1901 genes we generated 24 decisions (four classifiers for each of the six datasets) and obtained 24*16 base models. It should be noted that not only identifying the true functional category for a gene is important but also identifying if a gene does not belong to any functional category is critical. If a gene can confidently be identified as not belonging to a functional class, it will lead to a lot of savings in terms of further experimental investigation in many cases. Since it is also vital to quantify the performance on the prediction of the genes that belong to a functional category, we calculated the F-measure for evaluating and comparing the performance of different algorithms.

We observed that SVM performs better than the other classifiers in many cases. For SWsequence and Gasch datasets, SVM is outperformed by other methods like Adaboost (or in some cases NaiveBayes). This lead to the conclusion that due to the nature of the datasets and the properties of the base classifiers, different classifiers may perform well on different datasets. However, it is also observed that, though the classification accuracy varies between different functional categories, the performance of the base classifiers that work well on particular datasets does not vary significantly for different functional categories. This also suggests that it is the property of the dataset and the classifier that determines the effectiveness of the corresponding classifiers.

The ensemble that uses six SVMs as base classifiers performs better than the individual SVM classifier. We also observed a similar result for the ensembles that uses decision tree, NaiveBayes and Adaboost as base classifiers. This lead to the conclusion that ensemble can improve the performance to a certain extent. This outcome can be validated by the fact that these ensembles contain diversity by using different datasets and hence, the base classifiers used are diverse. However, since no base classifier of a particular type consistently outperformed all other types in all datasets; it is fair to assume that the performance variations of the base classifiers that we observe are primarily due to the property of the base classifier and the nature of the datasets. So, additional experiments were performed to choose the best base classifier for each of the datasets and an ensemble was built from those classifiers. This result is shown in Figure 1 along with the comparison with other majority voting approaches. Since we have already seen that the ensemble classifiers perform better than the corresponding classifiers, Figure 1 only reports the performance of the ensemble classifiers. One can observe that the ensemble from the best of the pools does not consistently outperform the other classifiers. In Figure 1, we compared all the aforementioned ensemble classifiers with our newly proposed decision boosting classifier. The proposed approach consistently outperforms the other ensemble classifiers and all the base classifiers. One of the possible reason for such performance improvement is because of the way the ensemble base classifiers are constructed and our proposed decision boosting method ensures the kind of dataset diversity that is needed for better performance. We compared our decision boosting approach for F-measure (see Table 5) with other ensemble techniques proposed by Re and Valentini (2010) for all the functional categories and depicted the superiority of the proposed DecBoost method (see Figure 2). Table 6 summarises the Win-Tie-Loss statistics. We observe that our approach consistently outperforms the decision templates.

Figure 1 Comparison of accuracy for different functional classes between different majority voting techniques and the proposed DecBoost (see online version for colours)

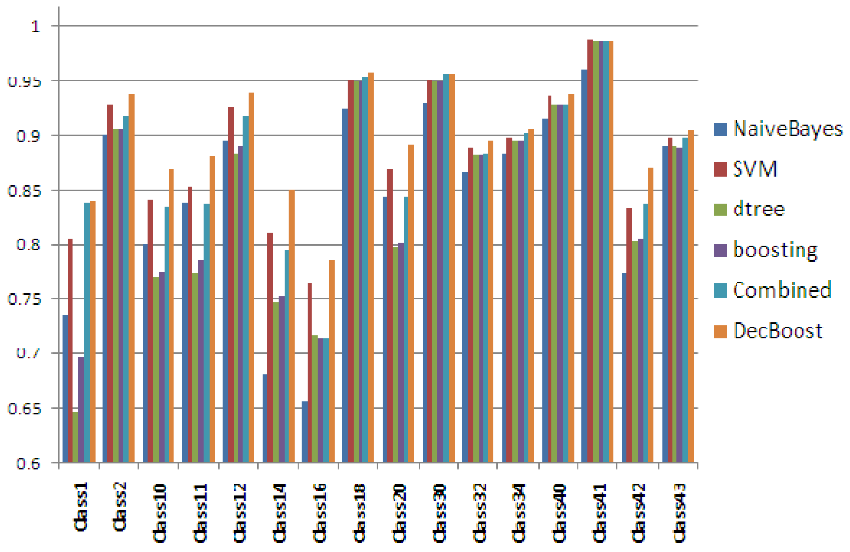

Figure 2 Comparison of F-measure values using different ensemble techniques and the proposed decision boosting for several FunCat classes (see online version for colours)

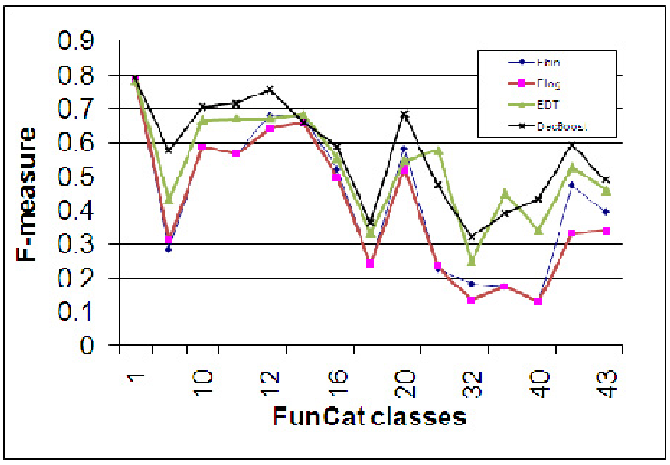

Table 5 Comparison of decision boosting to the other ensemble methods for different functional classes

FunCat class	*Ensemble$_{bin}$*	*Ensemble$_{log}$*	*Ensemble$_{DT}$*	*DecBoost*
1	0.7835	0.786	0.7845	0.792
2	0.2857	0.3125	0.4324	0.576
10	0.5887	0.5887	0.6666	0.705
11	0.5673	0.5673	0.6722	0.715
12	0.6814	0.6412	0.6715	0.758
14	0.6776	0.6581	0.6846	0.66
16	0.5217	0.4978	0.5543	0.59
18	0.2424	0.2424	0.3333	0.365
20	0.5828	0.5212	0.5465	0.686
30	0.2285	0.2352	0.5769	0.475
32	0.1842	0.1351	0.25	0.322
34	0.1764	0.1764	0.4509	0.391
40	0.1304	0.1304	0.3409	0.434
42	0.4736	0.3333	0.5279	0.594
43	0.3956	0.3414	0.46	0.49
AVG	0.4347	0.4111	0.5302	0.5702

Table 6 Win-tie-loss statistics for various ensemble methods

	Ensemble$_{DT}$	*Ensemble$_{lin}$*	*Ensemble$_{log}$*
DecBoost	12-0-3	14-0-1	15-0-0
Ensemble$_{DT}$	–	13-0-2	14-0-1
Ensemble$_{lin}$	–	–	7-5-3

We compared our method with late integration methods like Ensemble linear, Ensemble logarithmic, Ensemble Decision Templates as well as the early integration methods like VSI and Kernel Fusion (see Table 7). We observe that late integration generally outperformed the early integration methods and among the late integration methods, the proposed DecBoost algorithm consistently outperformed other state-of-the-art methods discussed earlier.

Table 7 Comparison of the proposed decision boosting method with different integration techniques proposed in the literature

Metric	*VSI*	*KF*	*Ensemble$_{log}$*	*Ensemble$_{DT}$*	*DecBoost*
F-measure	0.3213	0.3782	0.4111	0.5302	0.5702
Recall	0.226	0.3039	0.2974	0.4446	0.4634
Precision	0.653	0.6293	0.8443	0.7034	0.756

5.2 Integration results with all genes

Table 8 shows the results of different boosting methods on common and uncommon genes. We observe that DecBoost using common genes performs well on the common set of genes. However, for the uncommon genes, if we take only part of the model that can be fit with the particular genes, the results are not impressive. Using HeteroBoost, we observe the improvement of using all the genes during the training process on the uncommon genes. For the common genes, this model in some cases is outperformed by the common ensembles. Finally, *we observe that using the modified weighting criterion which emphasised the importance for uncommon genes more than the common genes gives performance improvement for the uncommon genes as well as the overall improvement.*

Table 8 Comparison of F-measure values in the presence of all the genes using heterogeneous boosting, decision boosting and boosting with only the common genes

Functional class	*Common genes*			*Uncommon genes*			*Overall results*		
	E_{CG}	E_{AG}	*HBOOST*	E_{CG}	E_{AG}	*HBOOST*	E_{CG}	E_{AG}	*HBOOST*
Metabolism	0.781	0.779	0.771	0.651	0.735	0.771	0.707	0.756	0.771
Energy	0.632	0.624	0.612	0.478	0.608	0.631	0.534	0.613	0.621
Transcription	0.712	0.690	0.673	0.609	0.683	0.721	0.653	0.687	0.695
Protein Synthesis	0.722	0.692	0.675	0.613	0.685	0.732	0.673	0.689	0.715
Protein Fate	0.691	0.688	0.683	0.591	0.638	0.706	0.654	0.664	0.699
Protein with Binding Function	0.619	0.622	0.631	0.509	0.601	0.651	0.559	0.613	0.645
Regulation of Metabolism	0.407	0.445	0.443	0.391	0.403	0.441	0.399	0.425	0.443
Cellular Transport	0.702	0.690	0.676	0.609	0.653	0.732	0.657	0.667	0.715
Cellular Communication/ Signal	0.515	0.520	0.503	0.410	0.453	0.551	0.456	0.487	0.535
Average	0.642	0.639	0.630	0.540	0.607	0.660	0.588	0.622	0.649

Notes: E_{CG}: ensemble with common genes; E_{AG}: ensemble with all genes; HBOOST: the proposed HeteroBoost method.

We also compared our heterogeneous boosting algorithm with kernel fusion methods with different imputation schemes to make it work for all genes (Tables 9 and 10). Table 10 shows accuracy values of the proposed method in comparison to different Kernel fusion methods. F-measure is a more appropriate metric than accuracy in this problem because it is more important to correctly associate the genes with a particular functional class than correctly detecting that the gene is not associated with other function class. In the previous section, we can observe that the kernel fusion method is outperformed by the ensemble techniques (Table 7) based on F-measure criteria. Table 9 shows that using the information of all genes using different imputation schemes improves the overall performance of the kernel fusion methods. However, despite such improvements, the result of kernel fusion methods with imputation is still inferior to our proposed heterogeneous boosting method. By the use of smoting on the training data, which is easily applicable and natural fits the boosting methods, the result of heterogeneous boosting significantly outperformed the kernel fusion method. It should be noted that, for different kernel imputation methods we do not see any consistent performance. On the other hand, the proposed heterogeneous boosting method consistently outperforms the kernel fusion methods. Since we do not see any significant and consistent difference in the performance between simple summation and weighted summation schemes, in Table 10 we only reported the weighted summation.

Table 9 Comparison of F-measure values for the proposed heterogeneous boosting method and different kernel fusion approaches

	Kernel fusion UMI		*Kernel fusion WSI*		*Kernel fusion NNI*		*Ensemble*
Functional class	Simple	Weighted	Simple	Weighted	Simple	Weighted	HBOOST
Metabolism	0.534	0.531	0.565	0.572	0.557	0.563	0.771
Energy	0.467	0.472	0.437	0.442	0.509	0.515	0.621
Transcription	0.473	0.482	0.422	0.418	0.495	0.467	0.695
Protein synthesis	0.515	0.509	0.519	0.523	0.607	0.593	0.715
Protein fate	0.509	0.513	0.515	0.509	0.544	0.567	0.699

Table 10 Comparison of accuracy values for the proposed heterogeneous boosting method and different kernel fusion approaches

Functional Class	*Kernel Fusion UMI*	*Kernel Fusion WSI*	*Kernel Fusion NNI*	*HBOOST*
Metabolism	0.783	0.789	0.773	0.783
Energy	0.771	0.783	0.783	0.819
Transcription	0.735	0.752	0.743	0.778
Protein Synthesis	0.756	0.752	0.743	0.793
Protein Fate	0.718	0.710	0.721	0.759

6 Conclusion

The availability of different sources of information on the same set of genes created new opportunities as well as challenges for the task of functional classification. Significant research efforts for robust integration of such abundant biological sources of information is being pursued at a rapid pace. For the task of functional classification of genes, integrating different sources at an early stage without the use of class information will provide significantly inferior result compared to those obtained from combining the decisions from multiple sources. In this paper, we developed a new method to combine the decisions of different classifiers in the individual dataset. Our study shows that different classifiers work well on different sources and their corresponding classification models have shown superior performance compared to individual classifiers. Our study also shows that boosting the decisions made from individual sources can obtain robust results on predicting gene functions. Furthermore, we proposed a novel integration framework based on re-weighting the uncommon genes helped in accurately predicting the overall functional classification. The proposed heterogeneous boosting method outperformed the standard kernel fusion-based approaches for integrating multiple sources in the presence of partial information in the context of gene function prediction.

Acknowledgements

This work was supported in part by the US National Science Foundation grants IIS-1231742 and IIS-1242304.

References

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

Aziz, M.S. and Reddy, C.K. (2010) 'Robust prediction from multiple heterogeneous data sources with partial information', *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, 26–30 October, Toronto, Ontario, Canada, pp.1857–1860.

Caruana, R. and Niculescu-Mizil, A. (2004) 'Data mining in metric space: an empirical analysis of supervised learning performance criteria', *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 22–25 August, Seattle, Washington, USA, pp.69–78.

Chawla, N., Bowyer, K., Hall, L. and Kegelmeyer, W. (2002) 'Smote: synthetic minority over-sampling technique', *Journal of Artificial Intelligence Research*, Vol. 16, pp.321–357.

Chechik, G., Heitz, G., Elidan, G., Abbeel, P. and Koller, D. (2008) 'Max-margin classification of data with absent features', *Journal of Machine Learning Research*, Vol. 9, pp.1–21.

des Jardins, M., Karp, P., Krummenacker, M., Lee, T. and Ouzounis, C. (1997) 'Prediction of enzyme classification from protein sequence without the use of sequence similarity', *Proceedings of the 5th International Conference on Intelligent Systems for Molecular Biology*, 21–26 June, Halkidki, Gteece, pp.92–99.

Dietterich, T.G. (2000) 'Ensemble methods in machine learning', *MCS '00: Proceedings of the 1st International Workshop on Multiple Classifier Systems*, 13–15 June, Seaside, CA, USA, pp.1–15.

Finn, R., Tate, J., Mistry, J., Coggill, P., Sammut, J., Hotz, H., Ceric, G., Forslund, K., Eddy, S., Sonnhammer, E. and Bateman, A. (2008) 'The Pfam protein families database', *Nucleic Acids Research*, Vol. 36, No. 281–288.

Freund, Y. and Schapire, R.E. (1996) 'Experiments with a new boosting algorithm', *International Conference on Machine Learning*, 3–6 Juky, Bari, Italy, pp.148–156.

Freund, Y. and Schapire, R.E. (1997) 'A decision-theoretic generalization of online learning and an application to boosting', *Journal of Computer Systems and Sciences*, Vol. 55, No. 1, pp.119–139.

Gasch, A.P., Spellman, P., Kao, C., Carmel-Harel, H., Eisen, M., Storz, G., Botstein, D. and Brown, P. (2000) 'Genomic expression programs in the response of yeast cells to environmental changes', *Molecular Biology of the Cell*, Vol. 11, No. 12, pp.4241–4257.

Graepel, T. (2002) 'Kernel matrix completion by semidefinite programming', *Proceedings of the International Conference on Artificial Neural Networks*, 28–30 August, Madrid, Spain, pp.694–699.

Karaoz, U., Murali, T., Letovsky, S., Zheng, Y., Ding, C., Cantor, C. and Kasif, S. (2004) 'Whole-genome annotation by using evidence integration in functional linkage networks', *Proceedings of the National Academy of Sciences USA*, Vol. 101, No. 9, pp.2888–2893.

Kittler, J., Hatef, M., Duin, R. and Matas, J. (1998) 'On combining classifiers', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 20, No. 3, pp.226–239.

Krishnaraj, Y. and Reddy, C.K. (2008) 'Boosting methods for protein fold recognition: an empirical comparison', *IEEE International Conference on Bioinformatics and Biomedicine, BIBM'08*, 3–5 November, Philadelphia, PA, USA, pp.393–396.

Kuncheva, L., Bezdek, J. and Duin, R. (2001) 'Decision templates for multiple classifier fusion: an experimental comparison', *Pattern Recognition*, Vol. 34, No. 2, pp.299–314.

Kuncheva, L. and Whitaker, C. (2003) 'Measures of diversity in classifier ensembles', *Machine Learning*, Vol. 51, No. 2, pp.181–207.

Lanckriet, G., De Bie, T., Cristianini, N., Jordan, M. and Noble, W. (2004) 'A statistical framework for genomic data fusion', *Bioinformatics*, Vol. 20, No. 16, pp.2626–2635.

Little, R.J.A. and Rubin, D.B. (2002) *Statistical Analysis with Missing Data*, Wiley Series in Probability and Statistics, Wiley.

Palit, I. and Reddy, C.K. (2012) 'Scalable and parallel boosting with mapreduce', *IEEE Transactions on Knowledge and Data Engineering*, Vol. 24, No. 10, pp.1904–1916.

Polikar, R. (2006) 'Ensemble based systems in decision making', *IEEE Circuits and Systems Magazine*, Vol. 6, No. 3, pp.21–45.

Re, M. and Valentini, G. (2010) 'Integration of heterogeneous data sources for gene function prediction using decision templates and ensembles of learning machines', *Neurocomputing*, Vol. 73, Nos. 7–9, pp.1533–1537.

Reddy, C.K. and Park, J-H. (2011) 'Multi-resolution boosting for classification and regression problems', *Knowledge and Information Systems*, Vol. 29, No. 2, pp.435–456.

Rhee, S., Wood, V., Dolinski, K. and Draghici, S. (2008) 'Use and misuse of the gene ontology annotations', *Nature Review Genetics*, Vol. 9, No. 7, pp.509–515.

Roli, F., Giacinto, G. and Gianni, V. (2001) 'Methods for designing multiple classifier systems', *Proceedings of the 2nd International Workshop on Multiple Classifier Systems, MCS'01*, 2–4 July, Cambridge, UK, pp.78–87.

Ruepp, A., Zollner, A.and Maier, D., Albermann, K., Hani, J., Mokrejs, M., Tetko, I., Guldener, U., Mannhaupt, G., Munsterkotter, M. and Mewes, H. (2004) 'The funcat, a functional annotation scheme for systematic classification of proteins from whole genomes', *Nucleic Acids Research*, Vol. 32, No. 18, pp.5539–5545.

Sharpe, P.K. and Solly, R.J. (1995) 'Dealing with missing values in neural network-based diagnostic systems', *Neural Computing and Applications*, Vol. 3, No. 2, pp.73–77.

Smola, A.J., Vishwanathan, S.V.N. and Hofmann, T. (2005) 'Kernel methods for missing variables', *Proceedings of International Workshop on Artificial Intelligence and Statistics*, 6–8 January, Barbados, pp.325–332.

Sonnenburg, S., Rätsch, G., Schäfer, C. and Schölkopf, B. (2006) 'Large scale multiple kernel learning', *Journal of Machine Learning Research*, Vol. 7, pp.1531–1565.

Spellman, P., Sherlock, G., Zhang, M., Iyer, V., Anders, K., Eisen, M., Brown, P., Botstein, D. and Futcher, B. (1998) 'Comprehensive identification of cell cycle-regulated genes of the yeast saccharomices cerevisiae by microarray hybridization', *Molecular Biology of the Cell*, Vol. 9, No. 12, pp.3273–3297.

Stark, C., Breitkreutz, B., Reguly, T., Boucher, L., Breitkreutz, A. and Tyers, M. (2006) 'Biogrid: a general repository for interaction datasets', *Nucleic Acids Research*, Vol. 34, pp.535–539.

Varma, M. and Babu, B.R. (2009) 'More generality in efficient multiple kernel learning', *ICML'09: Proceedings of the 26th Annual International Conference on Machine Learning*, 14–18 June, Montreal, Quebec, Canada, pp.1065–1072.

von Mering, C., Huynen, M., Jaeggi, D., Schmidt, S. and and B. Snel, P.B.P. (2003) 'String: a database of predicted functional associations between proteins', *Nucleic Acids Research*, Vol. 31, No. 1, pp.258–261.

Williams, D. and Carin, L. (2005) 'Analytical kernel matrix completion with incomplete multi-view data', *ICML Workshop on Learning with Multiple Views*, 7–11 August, Bonn, Germany, pp.80–86.

Zhao, X., Chen, L. and Aihara, K. (2008) 'Protein function prediction with the shortest path in functional linkage graph and boosting', *International Journal of Bioinformatics Research and Application*, Vol. 4, No. 4, pp.375–384.

Zien, A. and Ong, C.S. (2007) 'Multiclass multiple kernel learning', *ICML'07: Proceedings of the 24th international conference on Machine learning*, 20–24 June, Corvallis, OR, USA, spp.1191–1198.