View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Springer - Publisher Connector

BMC Bioinformatics

BioMed Central

Proceedings

Open Access

A Regression-based K nearest neighbor algorithm for gene function prediction from heterogeneous data

Zizhen Yao*[1] and Walter L Ruzzo[1,2]

Address: [1]Department of Computer Science and Engineering, AC101 Paul G. Allen Center, University of Washington, Seattle WA 98195, USA and [2]Department of Genome Sciences, University of Washington, Seattle WA 98195, USA

Email: Zizhen Yao* - yzizhen@cs.washington.edu; Walter L Ruzzo - ruzzo@cs.washington.edu

* Corresponding author

from NIPS workshop on New Problems and Methods in Computational Biology
Whistler, Canada. 18 December 2004

Published: 20 March 2006

BMC Bioinformatics 2006, **7**(Suppl 1):S11 doi:10.1186/1471-2105-7-S1-S11

Abstract

Background: As a variety of functional genomic and proteomic techniques become available, there is an increasing need for functional analysis methodologies that integrate heterogeneous data sources.

Methods: In this paper, we address this issue by proposing a general framework for gene function prediction based on the k-nearest-neighbor (KNN) algorithm. The choice of KNN is motivated by its simplicity, flexibility to incorporate different data types and adaptability to irregular feature spaces. A weakness of traditional KNN methods, especially when handling heterogeneous data, is that performance is subject to the often *ad hoc* choice of similarity metric. To address this weakness, we apply regression methods to infer a similarity metric as a weighted combination of a set of base similarity measures, which helps to locate the neighbors that are most likely to be in the same class as the target gene. We also suggest a novel voting scheme to generate confidence scores that estimate the accuracy of predictions. The method gracefully extends to multi-way classification problems.

Results: We apply this technique to gene function prediction according to three well-known *Escherichia coli* classification schemes suggested by biologists, using information derived from microarray and genome sequencing data. We demonstrate that our algorithm dramatically outperforms the naive KNN methods and is competitive with support vector machine (SVM) algorithms for integrating heterogenous data. We also show that by combining different data sources, prediction accuracy can improve significantly.

Conclusion: Our extension of KNN with automatic feature weighting, multi-class prediction, and probabilistic inference, enhance prediction accuracy significantly while remaining efficient, intuitive and flexible. This general framework can also be applied to similar classification problems involving heterogeneous datasets.

1 Background

With the exponential growth of genomic sequence data in the past decade, a major goal in the post-genomic era is to understand the biological functionality of genes and gene products. Computational methods have been proposed to infer functional annotation based on relevant biological information.

Methods based on sequence homology, such as BLAST, have been used for decades for effective gene annotation

in newly sequenced genomes. More recently, a variety of additional biological information sources have been used for functional prediction, such as gene co-evolution profiles [1], protein-domain fusion events [2], microarray expression data [3,4], protein interaction [5-7] and protein complex information [8]. However, even with the availability of the complete genomes of hundreds of organisms, application of any of these methods in isolation still leaves a large fraction of genes unannotated.

An important next step is to combine different data sources for functional prediction. Several statistical methods have been proposed as general frameworks to integrate heterogeneous datasets. Deng *et al*. [9] use a maximum likelihood method based on Markov random field (MRF) theory to predict protein function based on physical interactions, genetic interactions and gene expression. They found that the use of different sources of information indeed improved prediction accuracy when compared to using only one type of data. In application to yeast protein function prediction based on the MIPS classification scheme, they achieved 87% sensitivity at 57% specificity. One limitation of their technique is that the model is based on binary pairwise relations only. For continuous data such as gene expression measurements, there might be a loss of information due to discretization. Support vector machines (SVM) have also been extended to support heterogeneous data sets. Pavlidis *et al*. [10] suggest three ways to combine the data sets for SVM: early integration concatenates the feature vectors from each data source, intermediate integration adds the corresponding kernel matrices, and late integration adds the discriminant values from the SVM learned from each data source. More recent work by Lanckriet *et al*. [11] suggests a method that combines multiple kernels in an optimal fashion. They formulate the problem of combining kernels as a convex optimization problem that can be solved using semi-definite programming. Compared to the MRF method, they showed an improvement of ROC score from 0.715 to 0.851.

In this study, we suggest K-nearest-neighbor (KNN) methods as an alternative to solve this problem. In spite of their simplicity, KNN methods are among the best performers in a large number of classification problems. Because KNN methods do not make any assumption about the underlying data, they are especially successful when the decision boundaries are irregular, or a class has multiple prototypes. The flexibility of KNN is of great advantage for gene function classification problems, wherein the class boundaries are inherently vague, and many classes can not be categorized by a simple model.

The main idea of the classical KNN is the following: First design a set of numerical features to describe each data point, and select a metric, e.g. Euclidean distance, to measure the similarity of data points based on all features. Then for a target point, find its k closest points in the training samples based on the similarity metric, and assign it to a class by majority vote of its neighbors. A main weakness of this technique is that performance is subject to the often *ad hoc* choice of similarity metric, especially for heterogeneous datasets from which the derived features are of different types and scales, and are correlated. In addition, the standard KNN methods suffer from the curse of dimensionality. The neighborhood of a given point becomes very sparse in a high dimensional space, resulting in high variance.

Our proposed approach modifies the standard paradigm. Instead of building a single "global" similarity metric from all data sources, we design several "base" similarity measures between pairs of data points, one measure from each data source. Then we aim to estimate the likelihood of a pair being in the same class given the base similarities. In this formulation, the likelihood estimation can be approached as a classical regression problem. (See Equation 1 in section 2.1). The estimated likelihood function serves as the global similarity measure to choose the k nearest neighbors as before.

A key benefit of above approach is that heterogeneous data sources are weighted, and their correlation is taken care of automatically by regression, so that one can design one similarity measure at a time from a single data source and ignore the relationship among them. Our method is in a sense similar to the method of combining kernels due to Lanckriet *et al*. [11]. Each similarity measure in KNN corresponds to an SVM kernel, except that the former does not need to be semi-definite, thus allows greater design flexibility. In Lanckriet's method, the kernel weighting is integrated with the goal of maximizing SVM classification margin. In our framework, the base similarity measures are weighted to locate the "best" k nearest neighbors, which are mostly likely to share the same class as the target point, providing the most accurate prediction. This technique also helps to alleviate the curse of dimensionality by shrinking the unimportant dimensions of the feature space, bringing more relevant neighbors close to the target point.

Additionally, the probabilistic interpretation of the inferred similarity measure suggests a new voting scheme that facilitates multi-way classification and statistical inference. Instead of producing a single "best guess" for each gene, our voting scheme (see Equation 2 in section 2.2) generates a ranked list of all possible predictions with corresponding confidence scores, which provide good estimates of expected accuracy.

In summary, we propose a conceptually simple, computationally efficient method for integrating heterogeneous data sets for gene function prediction. We applied this method to the problem of gene function prediction in *E. coli* using information extracted from gene microarray expression data and sequence data. Results show that this technique significantly outperforms the naive KNN method, and combining all the data sources yields considerable improvement compared to using each data source alone. When compared to the SVM algorithm for integrating heterogeneous data, we achieved very competitive results (66.8% sensitivity at 50% accuracy and ROC score of 0.884 as compared to SVM with 67.6% sensitivity at 50% accuracy and ROC score of 0.915). Even though our method is slightly worse in terms of ROC score, it performs almost identically to SVM at low false positive rates. Similar conclusions apply to the classification results based on COG and Multifun classification schemes.

2 Methods and Data

Our algorithm can be briefly summarized as follows: In the training phase, we compute the base similarity measures from all genes in the training set, and combine them into a global similarity measure using a regression method. In the classification phase, for a gene with unknown functional classes, we choose its k nearest neighbors in the training set according to the trained similarity measure, and then use a customized voting scheme to generate a list of predictions with confidence scores. The details of the two major components of our algorithm – the similarity metric and voting scheme – are described below.

2.1 Regression Derived Similarity Metrics

The goal of the training phase is to construct a similarity metric that helps to locate the "best" neighbors -the neighbors that are most likely to be in the same class as the target. We do this as follows:

For every gene pair $p = (g_i, g_j)$ in the training set, we compute the value of the base similarity measure f_p^k from the kth data source. For example, f_k can be the correlation of expression profiles of two genes:

$$f_p^k = correlation(g_i.expr, g_j.expr)$$

We define a class sharing indicator C for each gene pair p, such that $C_p = 1$ if the two genes in p are in the same functional class, otherwise $C_p = 0$. The core of the algorithm is to find a function h such that

$$Pr\left\{C_p = 1 \middle| f_p^1, f_p^2, \ldots, f_p^m\right\} = h\left(f_p^1, f_p^2, \ldots, f_p^m\right) \quad (1)$$

In other words, we try to estimate the likelihood a pair of genes being in the same class as a function of a set of base similarity measures, which serves as the global similarity metric to choose the nearest neighbors. In the ideal world, this metric would assign value 1 for all pairs in the same class, and value 0 for pairs in different classes. Based on this metric, the nearest neighbors would be in the same class as the target gene, providing accurate predictions. In practice, however, due to the limitations of both data and methods to fit the function h, the learned likelihood function is only an estimate. Nevertheless, it integrates the base similarity measures in a way that is optimal in the KNN framework. Fitting function h falls into the realm of classical regression problems. We have applied two regression algorithms – logistic regression and local regression, to solve this problem.

Logistic regression is targeted at classification problems with binary or categorical response [12]. Let $X_p = \left(f_p^1, f_p^2, \ldots, f_p^m\right)$ be the feature vector, corresponding to the base similarity measures in our framework. The logistic model is described as the following:

$$\log\left(\frac{pr\left(C_p = 1 \mid X_p\right)}{pr\left(C_p = 0 \mid X_p\right)}\right) = \beta^T X_p + \alpha \quad (2)$$

where β is the weight vector, and α is a constant. Maximum likelihood estimates of α and β can be computed efficiently since the log likelihood function is concave. The learned weight vector β indicates the relative importance of each feature, and the probability function is used as the similarity metric. The linear model can be extended to quadratic functions, possibly with interaction terms to capture correlations between the features.

Local regression is another option to solve this problem [13]. In a nutshell, this technique estimates the regression function by fitting a different but simple model (e.g. a polynomial function) separately at each target point. Only the observations that are close to the target point are used to fit the model, and are weighted by their distances to the target point. Compared to logistic regression, local regression models provide greater flexibility, as the regression curve can approximate any smooth function. This method can also capture the correlation between the features naturally by fitting each model in a local region defined jointly by all the features. On the other hand, local regression is more computationally expensive, and less scalable. To fit local models, it requires a relatively dense neighborhood at every sample point. Local regression may also have troubles with discrete features and boundary data points.

We used the Splus [14] implementation of regression methods. The "glm" method was used for logistic regression with "family" parameter set to "binomial". By default, we used quadratic model with interaction terms, but the results based on linear model are very similar. For local regression, we used the "loess" method using quadratic functions and default span parameter.

2.2 Voting Scheme

A gene can belong to multiple function classes. Therefore, it is not sufficient to choose the maximum weighted prediction for a target gene, but rather, we need to assess the likelihood of all predictions. Our voting scheme is designed to provide a probabilistic measurement of the prediction quality.

Given the trained similarity metric, we can select the k nearest neighbors for a target gene g_i to form its neighborhood $N(g_i)$. Let g_j be a gene in $N(g_i)$, C_j a set of its function classes, and P_{ij} the similarity score between g_i and g_j, which estimate the probability that a prediction of g_i suggested by g_j is correct. To integrate the predictions suggested by all neighbors, we define the score for a prediction of function class C for g_i as the following:

$$P(C \in C_i) = 1 - \prod_{j:C \in C_j \wedge g_j \in N(g_i)} (1 - P_{ij}) \qquad (3)$$

We refer to the above score as the confidence score, which estimates the likelihood that the prediction is correct. Intuitively, if we assume the independent correctness of all predictions, then the confidence score computes the probability that at least one of the predictions suggested by neighbors of g_i with class C is correct, which is 1 minus the probability that all such predictions are wrong. The independence assumption is clearly inaccurate, thus we refer to this measure as a score instead of a probability.

Nevertheless, in practice, we found it quite satisfactory, assigning higher scores to predictions suggested by more neighbors, or neighbors with higher credibility. As we will show in the Results section, the confidence score is closely correlated with prediction accuracy.

The neighborhood size we chose is 10. Our experiments showed that the performance is stable with varying neighborhood size from 5–15, and degrades when the neighborhood size is too small or too large. When the neighborhood is too small, close neighbors with similar scores have to compete with each other to be selected. If the neighborhood is too large, confidence scores tend to inflate due to irrelevant neighbors.

Table 1: KEGG Functional Categories. Top level KEGG functional classes, with the number of genes in each class. (Jan. 2004)

Index	Class name	Size
1	Carbohydrate Metabolism	245
2	Energy Metabolism	163
3	Lipid Metabolism	45
4	Nucleotide Metabolism	114
5	Amino Acid Metabolism	231
6	Lipid Metabolism	67
7	Metabolism of Complex Carbohydrates	96
8	Metabolism of Complex Lipids	59
9	Metabolism of Cofactors and Vitamins	169
10	Biosynthesis of Secondary Metabolites	24
11	Biodegradation of Xenobiotics	50
12	Transcription	73
13	Translation	106
14	Folding, Sorting and Degradation	51
15	Replication and Repair	87
16	Membrane Transport	409
17	Signal Transduction	92
19	Immune System	4
20	Cell Motility	59

2.3 Data

We tested the performance of the RB-KNN methods on three functional classification schemes for *E. coli*. They are KEGG (Kyoto Encyclopedia of Genes and Genomes [15]), COG (Clusters of Orthologous Genes [16]), and MultiFun [17]. For KEGG and COG, we used the top level classifications, which include 19 and 18 classes respectively. MultiFun is a multi-dimensional classification schemes in which a gene belongs to multiple functional classes according to different classification criteria. For clarity, we selected the category "Metabolism", and used 8 functional classes within this category. In the results discussion, most analyses are based on KEGG, while the conclusions are generally applicable to COG and MultiFun (See supplementary website [18] for details). The KEGG functional classes are described in Table 1. We chose four base similarity measures based on microarray expression data and genomic sequence information. The details are described below:

1. Expression correlation

We compiled data from 106 two-color DNA microarray hybridization experiments performed on *E. coli*. 56 of them are from the Stanford SMD database [19], and 50 are from the Wisconsin ASAP database [20]. The data are briefly described in Table 2. The gene expression data consist of log transformed expression level ratios. We chose genes that are included in all experiments, and missing values were estimated by KNN imputation [21]. We normalized the data so that the mean of measurements from each array is 0, and standard deviation is 1.

Table 2: Summary of Microarray Experiments. 106 microarray experiments of *E. coli* performed under different conditions. Experiments 1 56 were collected from Stanford SMD database, and 57 106 from WISC ASAP database.

Index	Condition	# Experiments
1 to 15	UV exposure	15
16 to 42	Trypophan metabolism	27
43 to 52	rng deletion / rne deletion	10
53 to 56	Irp regulation	4
57 to 106	WISC calibrated experiments	50

For each pair of genes, we calculated the Pearson correlation of their expression profiles. The relationship between the expression correlation of a pair of gene and the likelihood that they are in the same KEGG function class is shown in Figure 1. When expression correlation is below 0.6, the likelihood of a gene pair being in the same class is close to random chance. As the expression correlation increases above 0.6, the fraction of gene pairs in the same class increases dramatically. When correlation is greater than .9, about 83% of the gene pairs are in the same class.

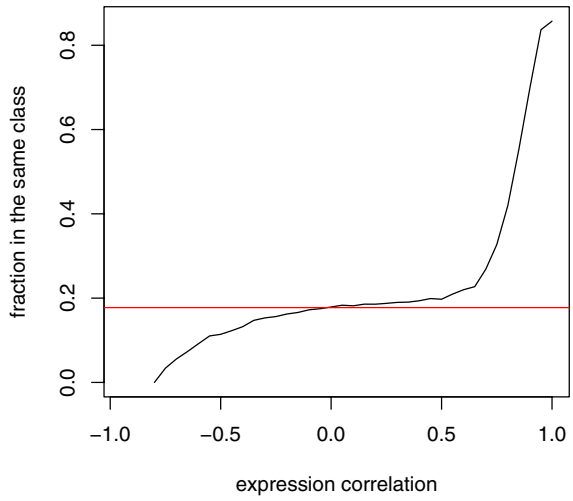

Figure 1
Expression Correlation Analysis. The expression correlations of gene pairs are divided into 40 bins within the range of (-1,1) with width 0.05. The y axis shows the fraction of pairs in the same function class in each bin. The red horizontal line shows the probability of a pair of genes being in the same class by chance (0.177).

2. Chromosomal distance

Chromosomal proximity information has been utilized to discover putative functional linkages between genes for Prokaryotes. When two genes appear near each other in the genomes of several distantly related organisms, it suggests the possibility that two genes might be functionally related [22]. In this study, we have not incorporated comparative genomic information. Nevertheless, we found that, about 70% of adjacent gene pairs on the chromosome of *E. coli* are in the same KEGG functional class. We define the chromosomal distance between a pair of genes to be the number of intermediate genes between them. Another distance metric we tried is the number of nucleotides between a gene pair, and we achieved similar results. To infer the regression model, we used the log of chromosomal distance.

3. Block indicator

Genes in the same operon tend to be involved in the same biological process. Thus operon structure provides another type of information that can be used for function prediction. Since operon information available at public data sources are usually not complete, constantly updated, and some are predicted computationally, we decided to introduce a similarity measure that is stable, and can be easily obtained even for a newly sequenced organism. We define "block" to be a maximal stretch of contiguous genes transcribed in the same direction. For each gene pair, we compute a "block indicator" which specifies whether the two genes are in the same block. We observed that of adjacent pairs of genes on the chromosome, about 80% of the pairs in the same block are in the same KEGG functional class, while only 25% of the pairs in different blocks are in the same class. This demonstrates that the block indicator adds extra information about gene functional relationships to chromosomal distance.

4. Paralog indicator

Liang *et al.* [23] showed that protein sequence similarity within an organism provides insight into function coupling. They clustered *E. coli* proteins into paralogous groups, and showed that members of the same paralogous group are usually functionally related. Protein sequences are compared pairwise using a dynamic algorithm implemented in the DARWIN [24] package. Protein pairs with PAW (accepted point mutations) scores above some threshold are chosen, and they are clustered by transitive closure. The list of paralogous protein families are available at [25]. According to this information, we introduced a similarity measure as the number of paralogous groups shared by a pair of genes. Note that a gene can belong to multiple paralogous groups, so the paralog indicator can be greater than 1, although it is mostly 0 or 1. We prefer this metric to the raw PAW score because it considers the cases in which proteins are indirectly related. Of 4666 pairs of paralogous proteins in this dataset, about 95% of them are in the same KEGG functional class.

2.4 Methods for Comparison

1. The naive KNN method

To evaluate the effectiveness of the regression methods for feature weighting and the voting scheme, we implemented a naive KNN method using Euclidean distance, scaled so that distance based on each individual data source has maximal value of 1. Given the k nearest neighbors, the naive KNN method selects the functional class that is voted for by the maximum number of neighbors. To break ties between the classes with the same number of votes, the one with minimum sum of distance from all supporting neighbors is chosen. We also applied the naive KNN method based on all combinations of data sources and compared it to the RB-KNN methods.

2. The SVM method

To compare RB-KNN to other methods for integrating heterogeneous data, we tested the SVM algorithm proposed in [10] on the same dataset. Among three strategies to combine multiple datasets – concatenate feature vectors, combine the kernels, or combine the discriminant values, we chose the strategy to combine the kernels, which Pavlidis *et al*. found to produce the best performance [10]. This method is in fact an un-weighted version of the method suggested in [11], which uses a semi-definite programming algorithm to determine the weights of each kernels. Even though the latter approach seems preferable, the corresponding software is not publicly available currently. Fortunately, the un-weighted method is nearly as effective as the weighted method in many cases, especially if all selected kernels are of comparable predictive power (personal communication, William Stafford Noble), which is true in our case.

We used the Gist software package for all SVM classifications, using 2-norm soft margin that accounts for the disparity in the number of positive and negative examples for each class. For the other parameters, we used the Gist default values. For details of Gist, see [26,27]. To perform multiway classification, we used a one-against-all strategy, training a SVM for each function class separately. We have defined 3 kernels based on expression data, block indicator and paralog indicator respectively. For expression data, we used the kernel $K(X, Y) = (X \cdot Y/(\sqrt{X.X}\sqrt{Y.Y})+1)^3$, which was shown in [10,28] to produce good performance on yeast function prediction. In comparison to the radial basis function (RBF) and linear kernel, this kernel performs very similarly to RBF on our dataset, both of which are significantly better than the linear kernel (The performance analysis of the linear kernel and RBF is available at the supplementary website [18]). The block indicator can be represented as the inner product of binary block vectors that specify the block membership of genes, thus it is a kernel itself. As with the block indicator, the paralog indicator is also a kernel since it can be represented as the inner product of the paralog vectors. To combine the three kernels, we add them after dividing the expression kernel by 8 so that all 3 kernels have similar scales. We did not use the chromosomal distance for SVM due to a couple of reasons. First, since the distance is defined on a circular chromosome, it is not obvious what an appropriate kernel is. We tried the empirical kernel map technique, using the inner product of the distance matrix row vectors as a kernel. However, the resulting SVM has very poor performance (ROC score = .581). Additionly, the test results for the RBKNN methods (Table 3) suggest that chromosomal distance is redundant given the other three data sources. Thus the SVM is not penalized for excluding this information.

3 Results

In this section, we will first compare the performance of RB-KNN with the naive KNN methods to demonstrate the strength of our techniques for feature weighting and voting. Second, we will compare our methods with a method for integrating heterogeneous data using SVMs. Third, we will give a performance analysis based on functional classes. We also include a brief discussion for the correlation between the confidence scores and prediction accuracy, and summary results based on all three classification schemes.

Finally, we will discuss some challenges of this problem based on analysis of the false predictions.

For convenience, all the KNN methods that we tested are named first by the algorithms, which can be **knn** (naive KNN method), **glm** (logistics regression based KNN) or

Table 3: Prediction Accuracy of naive Knn methods and RB-KNN methods. A combination of data sources is represented by a binary string, in which each bit specifies whether a data source is included, in the order of expression correlation, chromosomal distance, block indicator, and paralog indicator. glm refers the RBKNN method based on logistic regression, and loess refers to the RBKNN method based on local regression, loess is not applicable to "0001" (paralog indicator) and "0010" (block indicator) since the only data source is binary.

Combination	naive knn	glm	loess
0001	0.41	0.57	NA
0010	0.41	0.54	NA
0100	0.47	0.47	0.56
1000	0.40	0.44	0.52
1010	0.52	0.50	0.55
1011	0.61	0.68	0.68
1111	0.52	0.68	0.68

loess (local regression based KNN), followed by a specification of data sources. This specification is described as a binary string, in which each bit specifies whether a data source is included, in the order of **expression correlation, chromosomal distance, block indicator**, and **paralog indicator**. For example, loess.0001 means the local regression based KNN method using only the paralog indicator, while knn.1111 means the naive KNN method using all four data sources. We tested SVMs on the combined data with the expression, block and paralog information, which is referred to as "svm.comb". Since expression data alone are widely used to functional classification, we also tested SVMs in this case, named as "svm.exp". For each test, we performed 5-fold cross validation and reported overall performance.

3.1 Comparison of RB-KNN and the Naive KNN

To test the effectiveness of the RB-KNN methods for combining the heterogeneous data sources using the proposed feature weighting and voting techniques, we compared them with the naive KNN method. We tested each method using different combinations of data sources to study the predictive power of each individual data source, and whether combining more data sources improves the performance. Unless specified otherwise, the default classification scheme is KEGG. We only focused on the genes that are assigned to functional classes, 1603 in total. Each gene can belong to multiple functional classes, and the total number of function classifications is 2144.

The naive KNN method generates only one prediction for each gene, so for the sake of fair comparison, we also pick the single best prediction for each gene for RB-KNN methods. We noticed that the performance of the naive KNN is quite sensitive to k, the size of the neighborhood, and it generally favors small k.

Using all data sources, naive KNN has prediction accuracy of 52%, 50%, 46% and 40% for k = 1,3, 5,10 respectively. RB-KNN methods, however, are quite robust to the size of neighborhood. For example, the prediction accuracy for glm.1111 is 65%, 67.5%, 67.2% and 67.9% respectively for k = 1, 5,10,15. The prediction accuracies of naive KNN and RB-KNN methods are summarized in Table 3, with k = 1 for naive-KNN, and k = 10 for RB-KNN. Here, we only present the results for a few combinations of data sources. See Additional file 3 for the complete results. Table 3 shows that RB-KNN methods in general have better performance than naive KNN using the same data sources. Even based on a single data source, RB-KNN out-performs naive KNN because of the effectiveness of the voting scheme, which is better at combining predictions from the neighbors. The relative prediction power of each data source seems quite comparable. We also observed that using more data sources in the RB-KNN methods is almost always beneficial. In all our experiments, data set "1011" produces the best performance, which suggests that chromosomal distance is a redundant data source for functional prediction given the other three. By down-weighting the redundant information, RB-KNN methods have almost the same performance for data set "1111". Table 4 shows the weights for each data source in glm.1111. For naive KNN, however, adding chromosomal distance to the other three data sources results in significant decrease of performance. In general, the performance of loess is better than that of glm when only one or two data sources are used, but the difference in performance of the two methods becomes negligible as more data sources are used.

3.2 ROC Analysis of RB-KNN and SVM

We used the receiver operating characteristics (ROC) to compare the performance of RB-KNN and SVM algorithms. A ROC curve is a plot of false positive rate vs. sensitivity with variation of a threshold parameter. Since this is a multi-way classification problem, the total number of predictions is the product of the number of genes and the number of classes, which is 1603 * 18 = 28854 for KEGG. Each such prediction has an associated score, i.e., the confidence score in RBKNN, or the discriminate value in SVM.

All 28854 predictions are ranked based on their scores, and the predictions above a certain threshold are used to compute the false positive rate and senstivity for one point on the ROC curve. (One potential pitfall of using discriminant values to rank predictions is that they may not be comparable among different classes, as a distinct SVM is trained for each class. Plausibly, a preferable solution is to fit a probabilistic sigmoid function based on discriminant values for each class separately. We expected the resulting scores to be more comparable among different classes and produce better ranking, but the corresponding ROC score is actually worse, only 0.81 as compared to 0.91 of the former approach, so we did not take this

Table 4: Weights for each data sources computed by the RB-KNN method (glm.1111). The mean and standard deviations of weights of base similarity measures are presented based on the 5-old cross validation. For the ease of presentation, these weights are computed using the linear model, whose performance is slightly worse than the default quadratic model (accuracy of 0.65 as compared to 0.68).

Data sources	Expression correlation	Chromosomal distance	Block indicator	Paralog indicator
Weights	1.87 ± 0.12	0	1.68 ± 0.08	4.63 ± 0.13

approach.) Figure 2 shows the ROC curves of some of the algorithms we tested. Note that in this classification problem, the total number of false classifications is about an order of magnitude greater than the number of true classifications. Thus, even though the ROC scores of the algorithms are very close, in practice we are only interested in the region of these ROC curves where false positive rate is low. Within this region, the algorithms perform quite differently. The ROC curves show again that the predictive powers of all the data sources are quite comparable, ranked from the best to the worst in the order of paralog indicator, chromosomal distance, block indicator and expression correlation. Among the methods based on expression data, glm.1000 and svm.exp have very similar performance, and loess.1000 is significantly better. Similarly, loess.0100 has better performance than glm.0100 (data not shown) due to greater model flexibility.

Combining all four data sources results in significant improvement. At 50% accuracy, the sensitivities for glm.1111, glm.0001, glm.0010, loess.0100, loess.1000 are 66.8%, 55.1%, 46.9%, 50.4%, 41.3% respectively. Among all the methods that have been tested, svm.comb has the best performance with ROC score of 0.915. However, when the number of false positives is smaller than the number of true positives, as required by most applications, svm.comb has very similar performance as glm.1111. Its sensitivity at 50% accuracy is 67.6%, almost indistinguishable from the 66.8% of glm.1111.

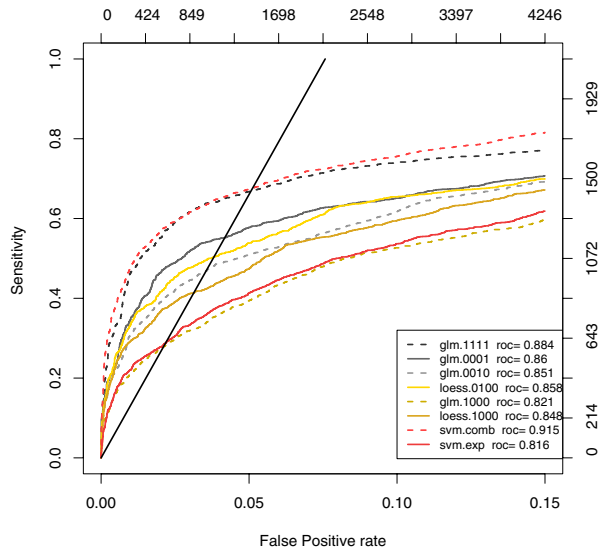

Figure 2
Comparison of ROC curves. The axis at the right side represents the number of true positives, and the axis at the top represents the number of false positives. The plot also shows the black straight line for TP = FP. The intersection of this line with each ROC curve indicates the corresponding sensitivity and false positive rate at 50% accuracy. We only plot the region where FP rate is less than 0.15.

3.3 Performance Analysis Based on Functional Classes

Figure 3 shows the sensitivities of four methods (svm.exp, loess.1000, glm.1111, svm.comb) at 50% accuracy for all functional classes. For each method, we divided all predictions based on the true functional classes of genes. Then for each class, we selected the predictions above a threshold that yields 50% accuracy, and calculated the percentage of genes that have been correctly predicted. We observed that for all classes except class 15, loess.1000 outperforms svm.exp quite dramatically. This is especially true in class 17, in which the sensitivity at the chosen threshold is 8.7% for svm.exp, and 71.7% for loess.1000. The SVMs based on linear or radial kernel perform poorly as well (0%, 10.9% sensitivity respectively).

This suggests that the decision boundary of class 17 has a particular structure that can not be captured by the chosen kernels for SVM, but can be by the RB-KNN methods. Across all functional classes, glm.1111 has very similar performance to svm.comb, both of which have superior performance compared to the methods based on expression data only. Class 20 (Cell Motility) and class 13 (Translation) are two classes with very good prediction accuracy using all 4 algorithms. The ROC scores of glm.1111 for these two classes are 0.97, and 0.96 respectively. The good classification results for these two classes can be attributed to the fact that most genes in these two classes are known to be co-regulated and many are transcribed in operons. Many members of Class 17 (Signal Transduction) and 16 (Membrane Transport) exhibit similar protein domain structure, thus the paralog indicator is very informative for both classes. The sensitivity of glm.0001 for these two classes is 80% and 76% respectively. In class 17, the use of paralog information by svm.comb helps to alleviate the poor performance of SVM based on the expression data, and its sensitivity reaches 84%, slightly worse than glm.1111.

3.4 Accuracy vs. confidence score

To study whether confidence score is a good estimate of prediction accuracy, we binned the confidence scores in range [0,1] into 20 intervals with width of 0.05, and computed the fraction of correct predictions in each interval. As shown in Figure 4, prediction accuracy is positively correlated with confidence scores, but the relationship is not linear. The non-linearity is due to the fact that in our voting scheme, the combined support from multiple neighbors is overestimated. Among 1116 predictions with

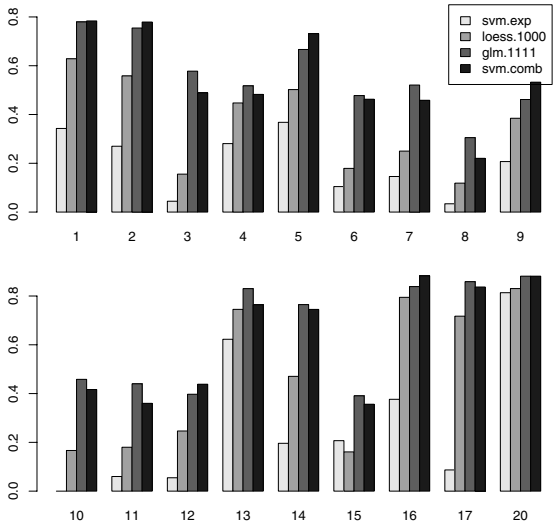

Figure 3
Sensitivity at 50% accuracy based on functional classes. The methods used for comparison include svm.exp, loess.1000, glm.1111 and svm.comb. The X axis shows the index of the functional class. For descriptions of the classes, refer to Table 1. The Y axis shows the sensitivity of each class at 50% accuracy.

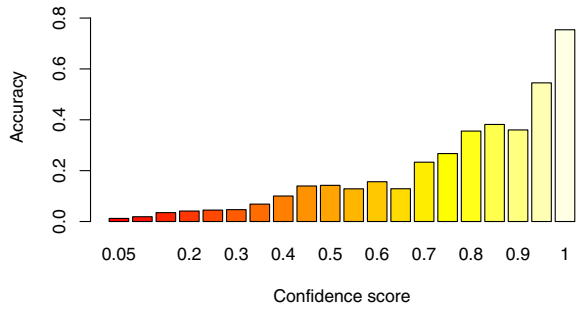

Figure 4
Confidence score vs. prediction accuracy. The method used for prediction is glm.1111. We divide the confidence scores into intervals of 0.05, and compute the prediction accuracy within each interval.

confidence score greater than 0.95, 80% (889) of them are correct.

3.5 ROC scores comparison for all 3 classification schemes

The performance analysis using ROC scores based on all three classification schemes are presented in Table 5. All the algorithms we tested have relatively poor performance on COG. COG is a classification scheme inferred by comparative genomic analysis, which is ignored in this study. We expect better results if such information is incorporated into the model. For Multifunc, we only include 8 functional classes, and each class is quite large. This is advantageous for an SVM, as it builds one model per class. It might be beneficial for RBKNN to construct a regression model from each class separately. (See Additional file 1, 2 for ROC curves for COG and Multifunc classification).

3.6 False predictions analysis

We analyzed a set of false predictions with high confidence score. Many of them arise from the situation in which genes belong to multiple functional classes. For such genes, different classification schemes may focus on different perspectives of their functional roles, and the annotations may be incomplete. In addition, since all genes interact with each other directly or indirectly, the class boundaries are vague. For example, the distinction between class 1 "Carbohydrate Metabolism" and class 7 "Metabolism of Complex Carbohydrate" is unclear. As another example, many genes in signal transduction (class 17) pathways are also involved in membrane transport (class 16).

We also analyzed the false negatives, i.e., true predictions with low confidence scores. Many false negatives are due to heterogeneous or small function classes. For example, class 10 (Biosynthesis of Secondary Metabolites) includes only 24 genes in our dataset, but it can be further categorized into 9 subclasses. Such classes are inherently difficult to categorize. Empty neighborhoods due to lack of information is another major contributor to false negatives. In KEGG, about 20% of genes have no neighbors with similarity score greater than .5, which suggests that they have no paralogs, co-expressed genes, or neighbors on the chromosome that are functionally annotated. This issue can only be resolved by adding more training data.

4 Conclusion and Discussion

The major contribution of this study is to propose a novel mechanism for combining heterogeneous data in the KNN framework, which includes two key components: a regression based weighting technique and a probabilistic voting scheme. The regression method determines the weight of each data source by considering their relative importance to function prediction and their correlations. The voting scheme facilitates statistical inference by integrating the function class nominations by the k nearest neighbors, and produces a ranked list of predictions with corresponding confidence scores. This technique also allows classification of a gene into multiple function classes. Our analysis showed that the local regression method has better performance with one or two data sources, presumably due to greater model flexibility, but

Table 5: ROC scores comparison based on three classification schemes

Methods	KEGG	COG	Multifun
loess.1000	0.821	0.794	0.837
loess.0100	0.858	0.791	0.842
glm.0010	0.851	0.784	0.842
glm.0001	0.86	0.766	0.809
glm.1111	0.884	0.807	0.863
svm.comb	0.915	0.829	0.889

logistic regression is more robust and scalable. We have demonstrated that using all four data sources yields the best performance. Compared to the Support Vector Machine algorithm that combines heterogeneous data, we achieved very competitive results. At accuracy greater than 50%, their ROC curves are almost identical. SVM tends to perform better for the genes that reside close to the class boundaries, but at the cost of a large number of false positives. At the same level of performance, RB-KNN methods have the advantage of simplicity: SVM requires kernel functions, which need to be semi-definite, while RBKNN accepts arbitrary pairwise relations. In addition, RBKNN produces supplementary information (e.g., the nearest neighbors with corresponding similarity values) to help interpretation of the results during post-processing.

Our classifier is based on microarray data and sequence data only, which are almost indispensable for any functional analysis. Therefore, our algorithm can be easily applied to other prokaryotes. For example, for a poorly annotated genome, we can apply comparative genomic methods such as BLAST to annotate the genes with homologs in other species, then apply RB-KNN to predict the function of the rest of the genes. For eukaryotes, operons are absent or rare, and chromosomal proximity tends to be less informative for functional coupling, therefore we would need to choose a different set of data sources.

Our methods can be improved in a number of ways. In this study, we built a single regression model for all samples in the training set. However, prediction power of each data source may vary from class to class, and they could be weighted differently. One advantage of our current approach is the abundance of training samples; one drawback is obviously the lack of descriptive power. The opposite is true of a one-class-one-model approach. An intermediate approach is to build class specific models for classes with large populations.

When features are highly correlated, the regression model can be very unstable. Principle component decomposition or regulated regression techniques such as lasso [29] or ridge regression [30] can be applied to generate more robust models. Another possible extension is to use more elaborate regression models. For example, we only extracted one base similarity measure, expression correlation, from 106 microarray experiments. However, some arrays maybe more informative than the others, and the correlation can vary significantly based on different sets of experiments. Such information is ignored by using Pearson correlation based on of all 106 experiments. One solution is to partition the experiments into several subsets, and derive a base similarity measure from each subset of experiments. They can be integrated directly with other data sources such as chromosomal proximity, or they can be merged by regression methods, which in turn can be integrated with other information. The benefit of a hierarchical model is that each regression involves only a few terms, so it is more robust than the fiat model which includes all the features.

We have shown that confidence scores are positively correlated with prediction accuracy, but the relationship is not linear, and the scores are inflated as the neighborhood size increases. For better quality control, we would prefer a scoring system that is more consistent with accuracy.

Finally we would like to incorporate more biological information such as evolutionary history and protein interaction into the model and apply it to other organisms. Our initial experiments showed good results on this difficult problem even with the most basic biological information, and we believe that RB-KNN is a good framework to tackle similar classification problems involving heterogeneous datasets.

Authors' contributions

LR and ZY conceived of the study. ZY performed the data collection, data analysis, and drafted the manuscript under the guidance and supervision of LR. All authors read and approved the final manuscript.

Additional material

Additional File 1

Comparison of ROC curves for COG. *The axis at the right side represents the number of true positives, and the axis at the top represents the number of false positives. The plot also shows the black straight line for TP = FP. The intersection of this line with each ROC curve indicates the corresponding sensitivity and false positive rate at 50% accuracy. We only plot the region where FP rate is smaller than 0.15.*
Click here for file
[http://www.biomedcentral.com/content/supplementary/1471-2105-7-S1-S11-S1.pdf]

Additional File 2

Comparison of ROC curves for Multifunc. *The axis at the right side represents the number of true positives, and the axis at the top represents the number of false positives. The plot also shows the black straight line for TP = FP. The intersection of this line with each ROC curve indicates the corresponding sensitivity and false positive rate at 50% accuracy. We only plot the region where FP rate is smaller than 0.20.*
Click here for file
[http://www.biomedcentral.com/content/supplementary/1471-2105-7-S1-S11-S2.pdf]

Additional File 3

Comparison of naive KNN methods and RBKNN methods on all combinations of data sources
Click here for file
[http://www.biomedcentral.com/content/supplementary/1471-2105-7-S1-S11-S3.txt]

Acknowledgements

We would like to thank William Stafford Noble for his suggestions on comparison experiments, Martin Tompa, Zasha Weinberg and Amol Prakash for reviewing the manuscript, and anonymous reviewers for insightful comments to improve this work.

References

1. Zheng Y, Roberts RJ, Kasif S: **Genomic functional annotation using co-evolution profiles of gene clusters.** *Genome Biol* 2002, **3(11):**.
2. Enright A, Iliopoulos I, Kyrpides N, Ouzounis C: **Protein interaction maps for complete genomes based on gene fusion events.** *Nature* 1999, **402:**86-90.
3. Pavlidis P, Lewis DP, Stafford W: **Exploring Gene Expression Data with Class Scores.** *Proceedings of the Pacific Symposium on Biocomputing* 2002.
4. Eisen M, Spellman P, Brown P, Botstein D: **Cluster analysis and display of genome-wide expression patterns.** *Proc Natl Acad Sci USA* 1998, **95:**14863-14868.
5. Fellenberg M, Albermann K, Zollner A, Mewes H, Hani J: **Integrative analysis of protein interaction data.** *Proc Int Conf Intell Syst Mol Biol* 2000, **8:**152-161.
6. Ge H, Liu Z, Church G, Vidal M: **Correlation between transcriptome and interactome mapping data from Saccharomyces cerevisiae.** *Nat Genet* 2001, **29(4):**482-6.
7. Grigoriev A: **A relationship between gene expression and protein interactions on the proteome scale: analysis of the bacteriophage T7 and the yeast Saccharomyces cerevisiae.** *Nucleic Acids Res* 2001, **29(17):**3513-9.
8. Gavin A, Bosche M, Krause R, Grandi P, Marzioch M, Bauer A, Schultz J: **Functional organization of the yeast proteome by systematic analysis of protein complexes.** *Nature* 2002, **415(6868):**123-4.
9. Deng M, Chen T, Sun F: **An integrated probabilistic model for functional prediction of proteins.** *Proceedings of the RECOMB* 2003:95-103.
10. Pavlidis P, Weston J, Cai J, Grundy WN: **Gene functional classification from heterogeneous data.** *RECOMB* 2001:249-255.
11. Lanckriet GRG, Deng M, Cristianini N, Jordan MI, Noble WS: **Kernel-based data fusion and its application to protein function prediction in yeast.** *Proceedings of the Pacific Symposium on Biocomputing* 2004:300-311.
12. Dobson AJ: *An Introduction to Generalized Linear Models London: Chapman and Hall*; 1990.
13. WS Cleveland EG, Shyu W: *Chapter 8 Local regression models Chapman and Hall*; 1991.
14. Splus: **S-Plus 6.0 for Unix User's Guide.** [http://www.insightful.com/products/splus/default.asp].
15. **KEGG: Kyoto Encyclopedia of Genes and Genomes** [http://www.genome.ad.jp/kegg/]
16. **COGs: Clusters of Orthologous Groups of proteins** [http://www.ncbi.nlm.nih.gov/COG/]
17. **MultiFun, a cell function assignment schema** [http://genprotec.mbl.edu/files/MultiFun.html]
18. Yao Z: **A Regression-based K Nearest Neighbor Method for Gene Function Prediction-Supplementary Website.** [http://bio.cs.washington.edu/yzizhen/RBKNN/].
19. **SMD: Stanford Microarray Database** [http://genome-www5.stanford.edu]
20. **ASAP: A systematic Annotation Package For Community Analysis of Genomes** [https://asap.ahabs.wisc.edu/annotation/php/ASAP1.htm]
21. Troyanskaya O, Cantor M, Sherlock G, Brown P, Hastie T, Tibshirani R, Botstein D, Altman R: **Missing value estimation methods for DNA microarrays.** *Bioinformatics* 2001, **7:**500-525.
22. Overbeek R, Fonstein M, D'Souza M, Pusch G, Maltsev N: **The use of gene clusters to infer functional coupling.** *roc Natl Acad Sci USA* 1999, **96(6):**2896-901.
23. Liang P, Labedan B, Riley M: **Physiological genomics of Escherichia coli protein families.** *Physiol Genomics* 2002, **9(1):**15-26.
24. GH G, Hallett MTKC, L B: **Darwin v.2.0: an interpreted computer language for the biosciences.** *Bioinformatics* 2000, **16:**101-103.
25. **GenProtec: E. coli Genome and Proteome Database** [http://genprotec.mbl.edu/]
26. Noble WS, Pavlidis P: **GIST.** [http://microarray.cpmc.columbia.edu/gist/].
27. Noble WS, Pavlidis P: **Gist: Support vector machine toolkit.** *Bioinformatics* 2004, **1(1):**1-3.
28. Brown MPS, Grundy WN, Lin D, Cristianini N, Sugnet CW, Furey TS, Jr MA, Haussler D: **Knowledge-based analysis of microarray gene expression data by using support vector machines.** *PNAS* 2000, **97(1):**262-267.
29. Tibshirani R: **Regression shrinkage and selection via the lasso.** *J Royal Statist Soc B* 1996, **8:**671-686.
30. Hoerl A, Kernnard R: **Ridge regression: Biased estimation for nonorthogonal problems.** *Technometrics* 1970, **12:**55-67.

Publish with **BioMed Central** and every scientist can read your work free of charge

"BioMed Central will be the most significant development for disseminating the results of biomedical research in our lifetime."

Sir Paul Nurse, Cancer Research UK

Your research papers will be:

- available free of charge to the entire biomedical community
- peer reviewed and published immediately upon acceptance
- cited in PubMed and archived on PubMed Central
- yours — you keep the copyright

Submit your manuscript here:
http://www.biomedcentral.com/info/publishing_adv.asp

BioMedcentral