BRIEFINGS IN BIOINFORMATICS. VOL 9. NO 3. 198–209
Advance Access publication January 11, 2008
doi:10.1093/bib/bbm064

ROC analysis: applications to the classification of biological sequences and 3D structures

Paolo Sonego, András Kocsor and Sándor Pongor

Submitted: 25th October 2007; Received (in revised form): 18th December 2007

Abstract

ROC ('receiver operator characteristics') analysis is a visual as well as numerical method used for assessing the performance of classification algorithms, such as those used for predicting structures and functions from sequence data. This review summarizes the fundamental concepts of ROC analysis and the interpretation of results using examples of sequence and structure comparison. We overview the available programs and provide evaluation guidelines for genomic/proteomic data, with particular regard to applications to large and heterogeneous databases used in bioinformatics.

Keywords: *protein similarity searching; classification; ROC analysis; performance assessment; function prediction*

INTRODUCTION

Automated classification plays a fundamental role in high-throughput projects such as genome sequencing and structural genomics. Thus, assessing the reliability of classifier algorithms has become essential to ensure data quality. There are excellent reviews on how to apply performance measures to classifiers in general [1], as well as in bioinformatics [2, 3]. Of the many performance measures and methods available, receiver operating characteristics (ROC) analysis [4, 5] occupies a special place. This is because, firstly, it provides a visual as well as numerical summary of a predictor's behavior in contrast to simple performance indices. Second, it is becoming a predominant method in bioinformatics applications. The data generated in bioinformatics—sequences, structures, etc.—are specific and quite different from the kind of data for which ROC analysis was originally invented. The goal of this review is to summarize its application to the specific tasks related to genome annotation, particularly to the classification of biological sequences and protein structures.

Originally developed for radar applications in the 1940s, ROC analysis became widely used in medical diagnostics, where complex and weak signals needed to be distinguished from a noisy background [6]. Subsequently, it gained popularity in machine learning and data mining [7, 8]. Fawcett [9] provides a good general introduction to the subject, and an extensive list of publications and tutorials can be found in [10]. Applications to bioinformatics were fostered by the seminal paper of Gribskov and Robinson [38]. The current popularity of ROC analysis in bioinformatics may be due to the visibly increasing use of machine learning techniques in

Corresponding author. Sándor Pongor, Protein Structure and Bioinformatics Group, International Centre for Genetic Engineering and Biotechnology, 34012 Trieste, Italy. Tel: +39-040 375 7300; Fax: +39-040 226 555; E-mail: pongor@icgeb.org

Paolo Sonego graduated in physics from the University of Bologna in 2003. Currently he is working for his PhD related to performance assessment of classification algorithms applied to biological data at The International Centre for Genetic Engineering and Biotechnology (ICGEB) in Trieste. His areas of research interest include supervised and unsupervised analysis, performance assessment of classification algorithms applied to biological data.

András Kocsor is with the Research Group on Artificial Intelligence of the Hungarian Academy of Sciences and University of Szeged, Hungary.

Sándor Pongor is with the Protein Structure and Bioinformatics Group, International Centre for Genetic Engineering and Biotechnology, 34012 Trieste, Italy and with the Bioinformatics Group, Biological Research Centre, Hungarian Academy of Sciences, Szeged, Hungary.

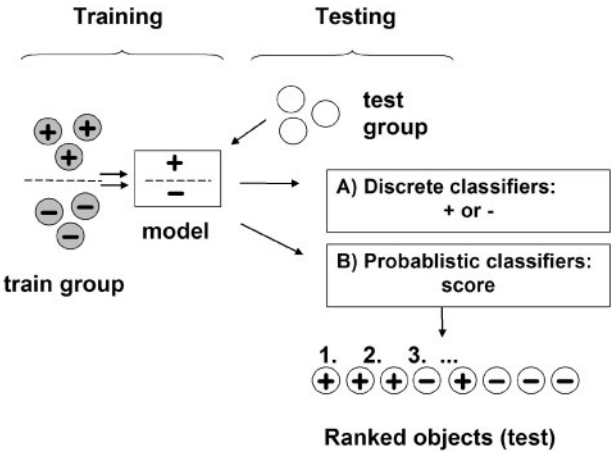
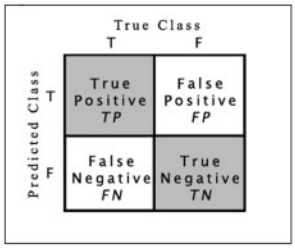


Figure 1: Binary classification. Binary classifiers algorithms (models, classifiers) capable of distinguishing two classes that are denoted + and −. The parameters of the model are determined from known + and − examples; this is the training phase. In the testing phase, test examples are shown to the predictor. Discrete classifiers can assign only labels (+ or −) to the test examples. Probabilistic classifiers assign a continuous score to the text examples, which can be used for ranking.



$$ACC = accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

$$SP = specificity = \frac{TN}{FP+TN}$$

$$TPR = sensitivity = \frac{TP}{TP+FN}$$

$$FPR = (1-specificity) = \frac{FP}{FP+TN}$$

Figure 2: The confusion matrix and a few performance measures that can be derived from the the number of true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN) in a test set. TPR is the true-positive rate or sensitivity, and FPR is the false-positive rate. A ROC curve is a TPR versus FPR plot.

computational genomics. Due to this sequence of events, current bioinformatics applications of ROC analysis use concepts and approaches taken from a variety of fields. This work seeks to provide an overview of ROC analysis as applied to molecular biology data. The rest of this review is structured as follows: (i) principles of ROC analysis, including comparison methods on biological databases; (ii) evaluation scenarios used in bioinformatics applications; (iii) available software and (iv) recommendations and caveats.

THE ROC PRINCIPLE

The fundamental use of ROC analysis, covered in this review, is its application to binary (or two-class) classification problems. A binary classifier algorithm maps an object (for example an un-annotated sequence of 3D structure) into one of two classes, that we usually denote as + and −. Generally, the parameters of such a classifier algorithm are derived from training on known + and − examples, then the classifier is tested on + and − examples that were not part of the training sets (Figure 1).

A *discrete classifier* predicts only the classes to which a test object belongs. There are four possible outcomes: *true positive*, *true negative, false positive, false negative*, schematically shown in Figure 2. If an object is positive and it is classified as positive, it is counted as a *true positive* (TP); if it is classified as negative, it is counted as a *false negative* (FN). If the object is negative and it is classified as negative, it is counted as a *true negative* (TN); if it is classified as positive, it is counted as a *false positive* (FP). If we evaluate a set of objects, we can count the outcomes and prepare a *confusion matrix* (also known as a *contingency table*), a two-by-two table that shows the classifier's correct decisions on a major diagonal and the errors off this diagonal (Figure 2, left). Alternatively, we can construct various numeric measures that characterize the accuracy, sensitivity and specificity of the test (Figure 2, right). These quantities are between 0 and 1 and can be interpreted as probabilities. For instance, the false-positive rate is the probability that a negative instance is incorrectly classified as being positive. Many similar indices are reviewed in [2] and [3].

Probabilistic classifiers, on the other hand, return a score that is not necessarily a *sensu stricto* probability but represents the degree to which an object is a member of one particular class rather than another one [11]. We can use this score to rank a test set of objects, and a classifier works correctly if the positive examples are at the top of the list. In addition, one can apply a decision threshold value to the score, for example, a value above which the prediction is considered positive. In such a way, we can change the probabilistic classifier into a discrete classifier. Naturally, we can select different threshold values, and, in this way, we can generate a (infinitely long) series of discrete classifiers for one probabilistic classifier.

Creating ROC curves

An ROC curve (Figure 3) is obtained by selecting a series of thresholds and plotting sensitivity on the

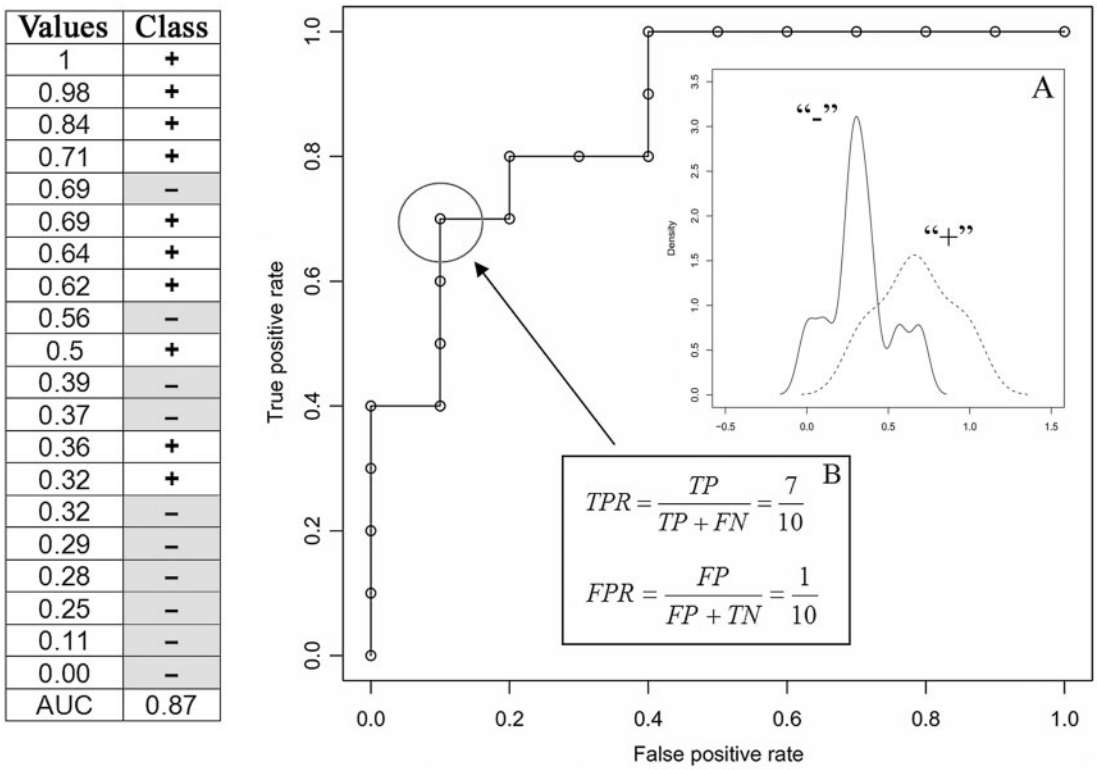


Figure 3: Constructing a ROC curve from ranked data. The TP, TN, FP and FN values are determined compared to a moving threshold; an example is shown by an arrow in the ranked list (left). Above the threshold, + data items are TP, − data items are FP. Therefore, a threshold of 0.6 produces the point FPR = 0.1, TPR = 0.7, as shown in inset B. The plot is produced by moving the threshold through the entire range. Data were randomly generated based on the distributions shown in inset A.

y-axis versus 1-specificity on the *x*-axis. Using the abbreviations of Figure 2, this is a TPR (true-positive rate) versus FPR (false-positive rate) plot. The output of our imaginary classifier is the ranked list shown on the left hand side of Figure 3. We can produce the ROC curve shown at the bottom left of the figure by varying a decision threshold between the minimum and maximum of the output values and plotting the FPR (1 − specificity) on the *x*-axis and the TPR (sensitivity) on the *y*-axis. (In practice, we can change the threshold so as to generate the next output value; in such a way, we can create one point for each output value). The empirical ROC curve generated for this small test set is a step function, and it will approach a continuous curve for large test sets.

Each point in this curve corresponds to a discrete classifier that can be obtained using a given decision threshold. For example, when the threshold is set to 0.6, the TPR is 0.7 and the FPR is 0.1. An ROC curve is thus a two-dimensional graph that visually depicts the relative trade-offs between the errors (false positives) and benefits (true positives) [11]. We can also say that an ROC curve characterizes a probabilistic classifier, and each point of this curve corresponds to a discrete classifier.

For sequences and 3D structure data, ROC curves are almost exclusively generated with the empirical (nonparametric) method outlined above [12]. In many other application areas, it is customary to use the so-called binormal approach [13–15] in which the two populations (classes) are assumed to be normally distributed. As a result, smooth ROC curves can be drawn using the sample means and variances of the two populations as shown in Figure 4. In bioinformatics applications, the empirical approach is preferred, since the data (such as BLAST similarity scores, 3D similarities) are known to be not normally distributed, and in many cases, a normal-scale transformation can not be performed in a reliable way. For a complete treatise of parametric and nonparametric methods see [16–18].

Interpretation of ROC curves

A ROC curve can be interpreted either graphically or numerically, as schematically shown in the inset

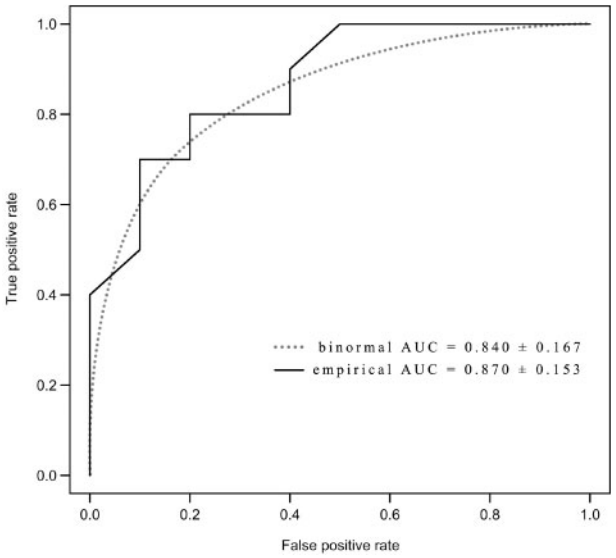


Figure 4: Binormal and empirical ROC curves.

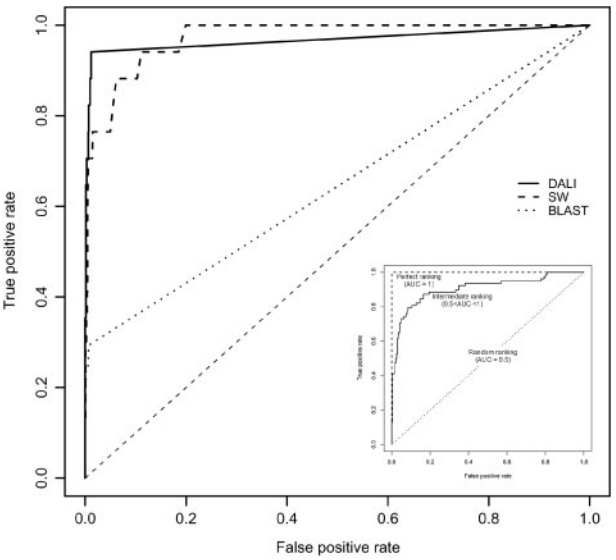


Figure 5: Examples of ROC curves calculated by pairwise sequence comparisons using BLAST [20], Smith Waterman [21] and structural comparisons using DALI [22]. The query was Cytochrome C6 from *Bacillus pasteurii*, the + group was composed of the other members of the Cytochrome C superfamily, and the − set was the rest of the SCOP40mini dataset, which were taken from record PCB00019 of the Protein Classification Benchmark collection [47]. The diagonal corresponds to the random classifier. Curves running higher indicate better classifier performance. Note that in this particular comparison, the performance of Smith Waterman approaches DALI. In a more thorough, database-wide comparison (Figure 6), DALI clearly outperforms the sequence comparison methods.

Table 1: Examples of AUC values for synthetic data[a]

Ranks	a	b	c	d	e	f	g	h
1	+	+	−	−	−	−	−	−
2	+	+	+	−	−	−	−	−
3	+	+	+	+	−	−	−	−
4	+	+	+	+	−	−	−	−
5	−	−	+	+	−	−	−	−
6	+	−	−	+	+	−	−	−
7	+	−	−	−	+	−	−	−
8	+	−	−	−	+	−	−	−
9	−	−	−	−	+	+	−	−
10	+	−	−	−	−	+	−	−
11	−	−	−	−	−	+	−	−
12	−	−	−	−	−	+	+	−
13	+	−	−	−	−	−	+	−
14	+	−	−	−	−	−	+	−
15	−	−	−	−	−	−	+	−
16	−	−	−	−	−	−	−	−
17	−	−	−	−	−	−	−	+
18	−	−	−	−	−	−	−	+
19	−	−	−	−	−	−	−	+
20	−	−	−	−	−	−	−	+
AUC:	0.87	1.00	0.93	0.88	0.70	0.50	0.30	0.00

[a]The numerical data are the same as used in Figure 1 (and shown here in column *a*). The data in columns *b*–*h* were generated by assigning the +/− values in the indicated manner.

of Figure 5. A perfect probabilistic classifier corresponds to the top ROC curve indicated by the hashed line. Such a classifier assigns higher scores to all positives than to any of the negatives, so the positives will be on top of the ranked list (Table 1, b). This curve is rectangular and its integral, the 'area under the ROC curve (AUC or AUROC), is equal to 1. The dotted diagonal line corresponds to a "random classifier"' that gives random answers, irrespective of the input. The integral (AUC value) of this curve is 0.5 (Table 1, f). A correct classifier has a ROC curve above the diagonal and an AUC> ~0.5. On the other hand, classifiers that give consistently the opposite predictions, ('anticorrelated' classifiers) give ROC curves below the diagonal and AUC values between zero and 0.5 (Table 1, g and h) [19]. Figure 5 shows a comparison of three real ROC curves obtained with BLAST, Smith Waterman and DALI [20–22]. Structural comparison with DALI performs better than the two sequence comparison methods; however, Smith Waterman does quite well on this particular task.

From a mathematical point of view, AUC can be viewed as the probability that a randomly chosen positive instance is ranked higher than a randomly chosen negative instance, i.e. it is equivalent to the

two sample Wilcoxon rank-sum statistic [23, 24]. Alternatively, AUC can be also be interpreted either as the average sensitivity over all FPRs or as the average specificity over all sensitivities [13]. Note that AUC_n values (described below under pairwise comparison) cannot be interpreted in this fashion.

In practice, AUC is often used as a single numerical measure of ranking performance. We note that ranking is dependent on the call distribution of the ranked set, so one cannot set an absolute threshold above which the ranking is considered good. In general, a high AUC value does not guarantee that the top ranking items will be true positives, as shown from the synthetic data in Table 1.

Averaging, confidence limits

The reliability of ROC curves can be estimated by repeating the analysis on a number of datasets and then calculating confidence bands from the resulting set of ROC curves [25]. The results can be presented as a bundle of curves (Figure 6) that can be aggregated into an average curve, and the confidence limits are given as the standard error calculated at various points on the curves. It is customary to calculate the average and variance of the AUC value in the same manner [26].

The +/− sets for repeated analyses can be produced by standard random subdivision techniques, such as bootstrapping, leave-one-out or k-fold cross-validation [1, 27]. However, random subdivision is known to give low variances for protein/gene classification tasks because once a member of a given family is known, the other members can be easily recognized [28, 29]. In order to get more realistic estimates on a classifier's generalization properties, the object classes can be subdivided into their known subgroups in such a way that, for instance, a protein superfamily is analyzed by one of its constituent families as the +test and the other families as the +train sets. This supervised cross-validation technique is designed to answer a biologically relevant question: how can a classifier recognize a new subtype based on the other known subtypes. Such datasets can be constructed for any database in which a hierarchical classification is defined, such as SCOP, CATH, COG or the taxonomic databases [29]. The bottleneck is the size of the test set, and it is recommended that one use groups in which +test $\geq$ 5 and +train $\geq$15 [24].

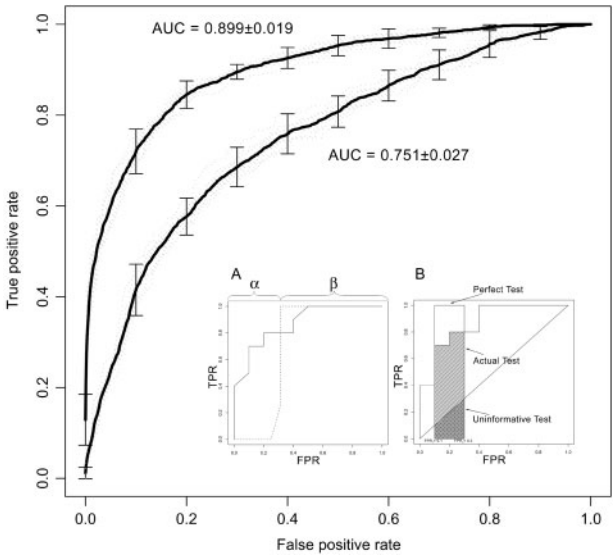


Figure 6: Averaging and comparison of ROC curves. Repeating the calculation on randomly sampled data from the same set can be used to generate a bundle of ROC curves that can be aggregated into average curves (bold line) with confidence intervals. The error bars in the figure correspond to 1.96 SD (95% confidence). The higher running curves and the larger AUC values correspond to better performance. Inset A: If ROC curves cross (data taken from Table 1, continuous line corresponds to column a, dotted line corresponds to column e), the results should be evaluated for the individual sections (such as indicated by α and β). Inset B: Illustration of partial AUC between FPR1 = 0.1 and FPR2 = 0.3.

Comparison of ROC curves

A comparison can be graphical or numerical, based on ROC curves or AUC values, respectively [25]. A ROC curve running above another is an indicator of better classifier performance, and by the same token, the bigger the AUC, the better the overall performance of the test. However, this reasoning is meaningful only if the two ROC curves do not cross at any point. If they do, then it makes intuitive sense to point out the region in which one classifier outperforms the other (Figure 6, inset B), but the comparison of the complete AUC values is not very informative [30].

A quantitative comparison of values is meaningful only if the variances are taken into account and significance can be estimated. Graphic comparison of ROC curves should be based on confidence bands [31] (Figure 6). The significance of the difference between two average AUC values can be estimated with a statistical test such as a two-tailed *t*-test or a

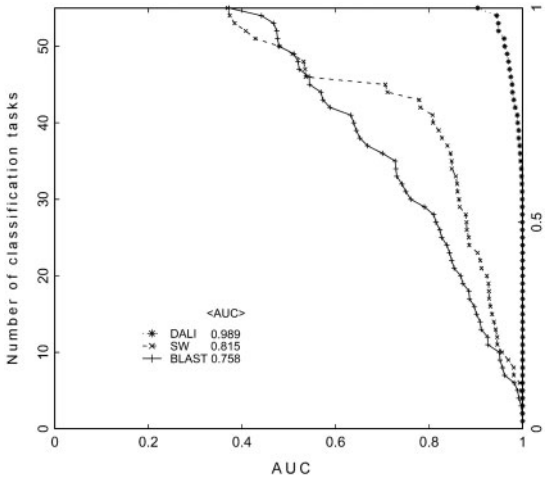


Figure 7: Database-wide comparison using cumulative AUC curves and similarity measures. The three methods are BLAST [20], Smith Waterman [21] and DALI [22], the comparison includes 55 classification tasks defined in the SCOP40mini dataset of the Protein Classification Benchmark [47]. The comparison was done by a nearest neighbor analysis using a groupwise scenario (Figure 7). Each graph plots the total number of classification tasks for which a given method exceeds a score threshold (left axis). The right axis shows the fraction of classification tasks (total number of tasks normalized to 1.00). The integral of the normalized cumulative AUC curve (which tends to the average of the constituent AUC values) is indicated for each method as <AUC>.

two-tailed signed rank test [32]. For a more critical comparison, the covariance of the two methods can be taken into account according to a modified z-test [33]:

$$z = \frac{| < \mathrm{AUC} >_1 - < \mathrm{AUC} >_2 |}{\sqrt{\mathrm{SE}_1^2 + \mathrm{SE}_2^2 - 2r\mathrm{SE}_1\mathrm{SE}_2}} \qquad (1)$$

where SE is the standard error of the respective mean AUC value, and r is the Pearson correlation coefficient calculated by the numerical values produced by the two classifiers for the same test set. The significance of z can then be calculated by standard methods [32]. DeLong and co-authors describe a nonparametric comparison of areas under two correlated ROC curves, which is implemented in various software packages [12].

Finally, we note that the graphical or numerical comparisons can also be restricted to selected parts of the ROC curve because the experimenter may want to 'zoom' on a range in which there is substantial difference between the two ROC curves to be compared (Figure 6, inset A). The partial area under the ROC curve (pAUC) [34] is defined as the area between two FPRs, FPR_1 and FPR_2, and can be denoted as $\mathrm{A}(\mathrm{FPR}_1 \leq \mathrm{FPR} \leq \mathrm{FPR}_2)$ [35] (Figure 6, inset B). Unlike AUC, whose maximum possible value is always 1, the magnitude of pAUC is dependent on the two FPRs chosen. pAUC can be normalized by dividing it by its maximum value, $(\mathrm{FPR}_2 - \mathrm{FPR}_1)$ [36]. The normalized pAUC or partial area index can then be interpreted as the average sensitivity for the range of FPRs or specificities chosen [36].

Database-wide comparisons

The application of machine learning classification algorithms to genomic data is a delicate task because the known object classes—such as those of domain-types and protein families—are highly variable in most of their characteristics (e.g. average sequence length, number of known members, strength of within-group similarity, overlap with neighboring groups, etc.). In order to make a reliable comparison of methods, we have to carry out a comparison on as many classes as possible, preferably with the supervised cross-validation scenario described earlier, which will estimate the generalization capability of the methods.

The reliability of a classifier on an entire database can be assessed from the arithmetic average of the AUC values for all classification tasks defined on a database as the overall performance indicator of a classifier for the given database. If we have many AUC values, we can also use a graphic comparison by plotting the number (or fraction) of AUC values above a certain threshold as a function of this threshold value [24]. In this plot, the higher curves indicate better performance (Figure 7). The numerical integral of the curve, sometimes called the cumulative AUC value, can serve as the overall performance indicator. It is easy to show that the numerical integral of the cumulative AUC curve (approximated with the rectangle rule) is the arithmetic average of the AUC values, so the two methods provide the same information. Liao and Noble [24] calculated the significance of database-wide comparisons using a two-tailed signed rank test between the individual AUC values calculated for the individual classification tasks [37].

APPLICATION SCENARIOS

As already mentioned, the standard application of ROC analysis is evaluating binary classifiers.

Table 2: Assessment of predictor performance by ROC analysis in various application areas

Representation	Predictor/Model	Ranking variable	Reference
Pairwise comparison			
Protein sequence	Nearest neighbor	Sequence alignment score	[38, 56, 57]
Protein structure	Nearest neighbor	VAST score, SHEBA score	[58, 59]
Generative models (consensus representations)			
Sequence group	Nearest neighbor	Sequence alilgnment score	[24, 60]
Multiple alignment	HMM		[40]
Discriminative models			
Vector of sequence alignment scores	SVM	Distance from hyperplane	[24, 47, 29]
Vector of structural similarity t scores (DALI, PRIDE)	SVM	Distance from hyperplane	[47]
Vector of sequence and structure similarity scores	Artificial Neural Networks	–	[47]
Vector of sequence-n-gram compositions	SVM		[61]

However, the practice in molecular biology has followed a different route, and ROC analysis was also applied to scenarios that are different from binary classification. In Table 2, we list the some of the typical applications along with the type of similarity/dissimilarity measure, the classifier algorithm (model) and the ranking variables they apply. The applications can be grouped into three broad categories, which are explained in greater detail subsequently.

Pairwise sequence comparison

Annotation by sequence similarity is a nearest neighbor scenario, i.e. a query is assigned to the functional/structural class of its most similar neighbor, with similarity determined with a program like BLAST. The first application of ROC analysis for sequence similarity searching dates back to the work of [38], who created a method capable of dealing with the scarcely annotated datasets available at the time (Figure 8A). Here, a separate ROC curve is created for each entry of the +test set. Originally, the authors used only a few members of a protein family as the +test, evaluated only a part of the top list by manually annotating the entries 'on the fly' as + or − and truncated the top list when a certain number of negatives were found. This number is indicated as an index, e.g. ROC_{50} means that the ROC analysis was performed on a top list that contained 50 negatives in addition to an unspecified number of positives. This technique is widely used today, and it is customary to compare methods simultaneously in terms of ROC_{10}, ROC_{20}, etc. We note that, while AUC for a complete test set is between 0.5 and 1.0, the AUC values for truncated top lists are between 0 and 1.0. In fact, the shorter the top list is, the smaller the AUC values [39].

Database-wide averages and variance values can be calculated by pooling the AUCs calculated for all the entries of the test set, and cumulative ROC curves can be drawn as outlined earlier.

Consensus descriptions (generative models)

Generative models provide a consensus description of an object group (the +train) and have been widely used for the description of protein families, domain types, etc. Methods including consensus sequences, frequency matrices, profiles and HMM descriptions fall into this category. In classification, the query is compared with the consensus description of a given class so as to provide a numerical score that can be used for ROC analysis (Figure 8B). Jaakkola and co-workers [40] use an HMM (trained from positive examples only) for mapping each new protein sequence on to a fixed length vector, and this vector is then passed to a support vector machine (SVM). SVM is a discriminative model (below), so this approach is actually a hybrid of generative and discriminative models.

Discriminative models

Finally, discriminative models are those in which the model depends both on the +train and −train groups. SVMs, artificial neural networks and other machine learning algorithms fall into this category. Their use follows the standard data mining scenario of binary classification for which ROC analysis was originally developed (Figure 1).

SOFTWARE FOR ROC ANALYSIS

Because ROC analysis has been used in medical diagnostics for many years, there are several easy to

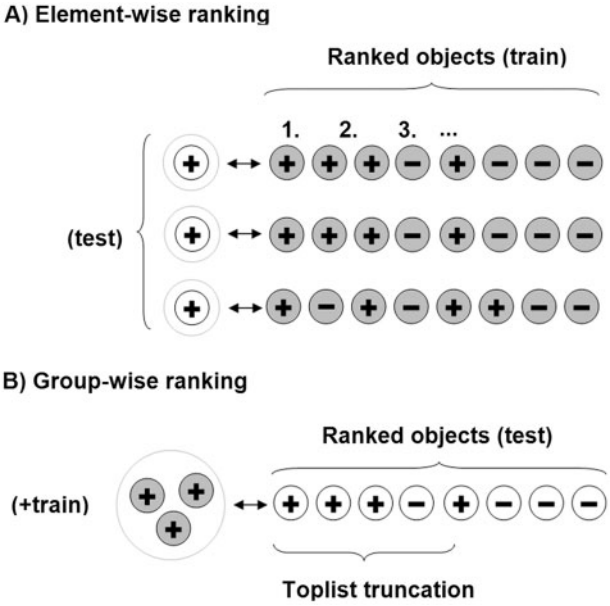


Figure 8: Ranking scenarios for calculating ROC. In the elementwise scenario (A), each query is compared to a dataset of + and − train examples. A ROC curve is prepared for each query and the integrals (AUC-values) are combined to give the final result for a group of queries. In the groupwise scenario (B), the queries of the test set are ranked according to their similarity to the +train group, and the AUC value calculated from this ranking is assigned to the group. Note that both A and B are one-class classification (outlier detection) scenarios because the negative group has no explicit influence on the ranking variable.

use, general purpose software packages available (for a review see [41, 42]). Some of the packages are open source, and users who want to try ROC analysis of small datasets can use either these or the ROC calculators on the web, such as the application available on the website of John Eng [43].

Users interested in a large number of ROC calculations, such as those necessary for database-wide comparisons, may prefer to use the facilities available in higher level programming environments for statistics analysis, such as R, WEKA and MATLAB®.

R is an open source statistical computing environment that contains numeric and graphic procedures for a large number of statistical methods [44], including several solutions for ROC analysis. The ROCR package [45] is perhaps the most versatile application within R, as it can create a ROC plot (as well as precision-recall curves and lift curves) and allows the use of confidence intervals based on bootstrapping, AUC and partial AUC values as well as a wide range of other performance measures. The extensive graphics capabilities and the availability of a bioinformatics framework such as Bioconductor [46] make R a good solution for academic bioinformatics researchers. The reference [47] describes a framework and provides downloadable R scripts for the RC analysis of various classifiers.

WEKA is an open source toolkit for machine learning designed for developing and testing algorithms for data mining [48]. It offers many state-of-the-art approaches in an object-oriented framework, including classifiers (SVMs, decision trees, rule learners, etc.) and clustering methods operated via both a graphical and command line interface. It includes several methods for classifier evaluation. Its ROC Analysis module uses a transformation analogous to Equation (2) as a default. WEKA can also interoperate with R via the RWeka package [49], which extends the capabilities of both environments. The BioWeka project [50] is designed to interface WEKA for bioinformatics applications.

The ROC Curve Toolkit for MATLAB®, which is available at [51], implements some of the basic methods for constructing and processing ROC curves as discussed in [11]. MATLAB® is a high-performance language, especially suited for problems involving matrix and vector formulations. The ROC Curve Toolkit for MATLAB® is well-suited to be used with other available machine learning tools.

RECOMMENDATIONS AND CAVEATS

General scope and potential drawbacks

ROC analysis is based on ranking a database of objects according to a similarity/dissimilarity measure. The ranking ability is a general property of the classification algorithm and not of a specific classifier that is a particular instance (parametrization) of the algorithm. For example, ROC analysis can be used to compare algorithms like BLAST and Smith Waterman on a given sequence database but will not tell anything about the thresholds to be applied. This means that a high AUC value will not guarantee that every classifier built using a given algorithm will be efficient; it only indicates that the algorithm is suitable for building good classifiers.

The framework of a classifier comparison includes (i) a similarity/dissimilarity measure, (ii) a classifier

algorithm that provides a ranking, as well as (iii) a dataset divided into classification tasks. Each of these elements can be analyzed separately, but in view of the extreme variability of protein and gene classes, it is recommended that one keeps as many conditions constant as possible. This common sense requirement has led to a comparison strategy in which the classification task (i.e. the +train, −train, +test, −test groups) is entirely predefined, so the calculations can be repeated on precisely the same tasks. A collection of pre-constructed datasets produced by various random and supervised cross-validation methods is now included into the Protein Classification Benchmark database [47]. In this collection, there is a number of classification tasks (+/− test and train groups) defined for each dataset, so the AUC for a given dataset can be calculated as the average of the average of the resulting AUCs.

ROC analysis is primarily meant for binary classifiers, i.e. for problems where + and − benchmark datasets can be safely identified [52]. If there are unknown and undecided cases, ROC analysis can only be applied to a safe, manually curated subset, and it may be questionable whether or not a subset is sufficiently representative of the variability of the entire database.

Comparing similarity measures

When the goal is to compare similarity/distance measures without explicit reference to a learning algorithm, ROC curves are often calculated from a nearest neighbor model in which a pairwise score is the ranking variable. One can use an element-wise scenario originally described by [38] or a group-wise scenario, which is a generative model in which the maximum similarity to the +train group is used as the ranking variable. Neither pairwise comparisons, nor generative models are *sui generis* binary classifications; rather, they resemble the framework of a one-class classification or an outlier detection [53] that is characterized by a closed decision surface, in contrast to the open detection surface of a binary classification. As a consequence, some of the fundamental conditions of ROC analyses are actually not met. A practical alternative applicable to groupwise evaluations is to transform a similarity scores into a likelihood ratio approximant:

$$L = \frac{S^+}{S^-} \qquad (2)$$

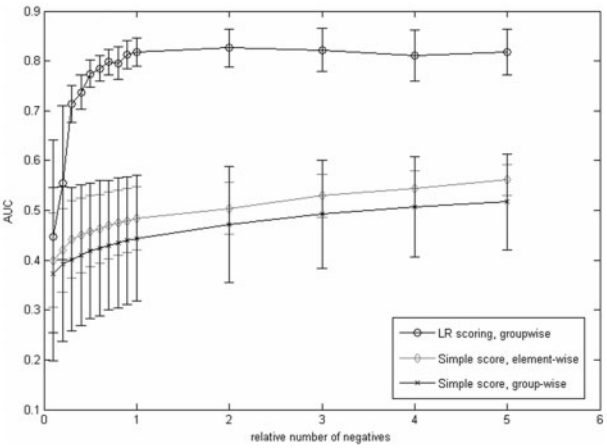


Figure 9: Comparison of various ranking and scoring scenarios calculated by varying the number of negatives in the ranking. Average AUCs were calculated for all 246 classifications tasks defined from the sequences taken from the SCOP database, compared with the Smith Waterman algorithm. The error bars indicate standard deviations calculated from the 246 tasks. This is a measure of dataset variability and not the evaluation. Note that the group-wise scenario with likelihood-ratio scoring gives values that are independent of the size of the negative set, while the results of the others show an increasing tendency [55].

where S^+ and S^- are the highest (i.e. nearest neighbor) similarity scores in the +train and −train, respectively [54]. This transformation makes the AUC values less dependent on class sizes, as shown in Figure 9.

Class imbalance problems and comparison of datasets

For training and evaluating binary classifiers, having the same number of positive and negative examples is recommended [1, 2]. This condition is practically never met in bioinformatics databases because we have many fewer positives than negatives. A correct solution would be to select randomly equal numbers from the two classes and construct many classifiers for all cases, which is clearly too time consuming if we need to repeat the training for each task. For this reason, there are very few studies in which bootstrapping is used for training classifiers for a comprehensive set of known categories. Instead, bioinformaticians use database-wide averages (such as average AUCs) to characterize and compare algorithms. Such an average value contains contributions from classes of various sizes, so it may be a good

indicator of how a classifier works in general, but it provides no information on how it behaves on particular classes. It is apparent that a transformation according to Equation (2) helps to decrease class imbalance problems, but this effect should be considered database-dependent.

Finding classes that are difficult to distinguish is an especially delicate task due to the large variability of proteins and genes. For instance, there are protein domain types in which only a few, well-conserved members are known, while other classes have several thousand, loosely conserved members. Calculating AUCs on truncated toplists, such as a ROC_{50}, may be quite misleading in these cases because the length of the lists and the number of positives and negatives included in the list may be very different in the two cases. One can partially circumvent this problem by the so-called balanced ROC (BAROC) protocol [55], in which the length of the toplist is proportional to the number of members in the positive. Plotting the AUC as a function of the positive/negative ratio in the toplist is also a useful tool for finding difficult groups in a large database. However, there are no general solutions that can be equally applied to very small and very large groups.

Which ranking scenario to use?

Generally speaking, the type of the algorithm will determine the type of ranking scenario that should be used. Discriminative models (artificial neural networks, SVMs) can be trained and tested according to the classical binary classifier scheme (Figure 1). Generative models can be tested according to a groupwise scenario, while simple (nearest neighbor) comparison methods can be tested either by the element- or the group-wise scenario. Figure 9 shows a comparison of the two scenarios on SCOP sequences and the Smith Waterman similarity score, using different class sizes. The results are comparable, but it is clear that one can quantitatively compare AUC values only if they are determined by the same scenario. We note that the some publications do not indicate clearly the scenario used, so comparison of published AUC values may be misleading. The time-requirements of the protocols also differ. In the elementwise evaluation, we have as many AUC calculations as there are test objects. With the groupwise scenario, we only have as many calculations as there are test groups.

CONCLUSIONS

ROC analysis provides statistically supported numerical and graphical tools for characterizing the performance of predictors. It can be used to investigate the performance of learning algorithms under changing conditions, such as misclassification costs or class distributions, and is thought to be more informative than simple numerical performance measures. Nevertheless, the variability of data used in computational genomics creates problems for model comparison, so it is recommended that algorithms be benchmarked on well-controlled datasets with standardized comparison protocols. It is important to note that ROC analysis characterizes the ranking potential of an algorithm and not the performance of an actual classifier. It is recommended that performance assessment should not to rely on a single method, such as ROC analysis, but should also include other comparison measures into the evaluation [2, 3].

Key Points

- ROC analysis is a visual as well as numerical method suitable for assessing the performance of classification algorithms in bioinformatics such as used for biological sequences and 3D structures.
- It shows how well the positive and negative samples are separated in a list ranked according to a given algorithm. This is a global measure which does not provide indications on how to construct a classifier for practical use.
- There are two major 'scenarios': element- and group-wise evaluation that give slightly different results.
- Specific methods (truncated toplists, supervised cross-validation, database-wide averaging) were developed, in order to deal with the highly variable and imbalanced datasets of biological sequences and 3D structures.
- Because of the high variability of genomic data, it is recommended that classification algorithms be benchmarked on controlled datasets and with well-documented comparison protocols.

Acknowledgements

The authors thank Luigino Dal Maso (Centro di Riferimento Oncologico, Aviano, Italy), Peter A. Flach (University of Bristol, UK), Róbert Busa-Fekete and Attila Kertész-Farkas (University of Szeged, Hungary) for their critical reading of the article and useful suggestions. Work at ICGEB was supported in part by grants from the Ministero dell'Università e della Ricerca [D.D. 2187, FIRB 2003 (art. 8), 'Laboratorio Internazionale di Bioinformatica']. A. Kocsor was supported by the János Bolyai fellowship of the Hungarian Academy of Sciences.

References

1. Duda RO, Hart PE, Stork DG. *Pattern Classification,* 2nd sub edn. New York: Wiley-Interscience, 2000.

2. Baldi P, Brunak S, Chauvin Y, *et al*. Assessing the accuracy of prediction algorithms for classification: an overview. *Bioinformatics* 2000;**16**:412–24.
3. Bajić VB. Comparing the success of different prediction software in sequence analysis: a review. *Briefings Bioinformatics* 2000;**1**:214–28.
4. Egan JP. *Signal Detection Theory and Roc Analysis*. New York: Academic Press, 1975.
5. Zweig MH, Campbell G. Receiver-operating characteristic (ROC) plots: a fundamental evaluation tool in clinical medicine. *Clin Chem* 1993;**39**:561–77.
6. Lusted LB. Signal detectability and medical decision-making. *Science* 1971;**171**:1217–9.
7. Provost F, Fawcett T. Robust classification for imprecise environments. *Mach Learn* 2001;**42**:203–31.
8. Flach P, Blockeel H, Ferri C, *et al*. Decision support for data mining: introduction to ROC analysis and its application. In: *Data Mining and Decision Support: Aspects of Integration and Collaboration*. Dordrecht, The Netherlands: Kluwer Academic Publishers, 2003, 81–90.
9. Fawcett T. An introduction to ROC analysis. *Pattern Recogn Lett* 2006;**27**:861–74.
10. ROC Links. http://www.tech.plym.ac.uk/spmc/links/roc/roc_links.html (11 October 2007, date last accessed).
11. Fawcett T. *ROC Graphs: Notes and Practical Considerations for Researchers*, 2004.
12. DeLong ER, DeLong DM, Clarke-Pearson DL. Comparing the areas under two or more correlated receiver operating characteristic curves: a nonparametric approach. *Biometrics* 1988;**44**:837–45.
13. Metz CE. Basic principles of ROC analysis. *Semin nucl med* 1978;**8**:283–98.
14. McClish DK. Comparing the areas under more than two independent ROC curves. *Med Decis Making* 1987;**7**:149–55.
15. Hanley JA. The use of the 'binormal' model for parametric ROC analysis of quantitative diagnostic tests. *Med Decis Making* 1988;**8**:197–203.
16. Goddard MJ, Hinberg I. Receiver operator characteristic (ROC) curves and non-normal data: an empirical study. *Stat Med* 1990;**9**:325–37.
17. Pepe MS. *The Statistical Evaluation of Medical Tests for Classification and Prediction*. USA: Oxford University Press, 2004.
18. Obuchowski NA. ROC analysis. *Am J Roentgenol* 2005;**184**:364–72.
19. Flach P, Wu S. Repairing concavities in ROC curves. *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI' 05)*. Menlo Park, CA, 2005;702–07.
20. Altschul SF, Gish W, Miller W, *et al*. Basic local alignment search tool. *J Mol Biol* 1990;**215**:403–10.
21. Smith TF, Waterman MS. Identification of common molecular subsequences. *J Mol Biol* 1981;**147**:195–7.
22. Holm L, Park J. DaliLite workbench for protein structure comparison. *Bioinformatics* 2000;**16**:566–7.
23. Hanley JA, McNeil BJ. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology* 1982;**143**:29–36.
24. Liao L, Noble WS. Combining pairwise sequence similarity and support vector machines for detecting remote protein evolutionary and structural relationships. *J Computat Biol* 2003;**10**:857–68.
25. Macskassy SA, Provost F, Rosset S. ROC confidence bands: an empirical evaluation. *Proceedings of the 22nd International Conference on Machine Learning*, 2005, 537–44.
26. Cortes C, Mohri M. Confidence intervals for the area under the ROC curve. *Advances in Neural Information Processing Systems* 2004;**17**.
27. Bandos AI, Rockette HE, Gur D. Resampling methods for the area under the ROC curve. *Proceedings of the ICML 2006 workshop on ROC Analysis in Machine Learning*, 2006.
28. Elofsson A, Sonnhammer E. A comparison of sequence and structure protein domain families as a basis for structural genomics. *Bioinformatics* 1999;**15**:480–500.
29. Kertész-Farkas A, Dhir S, Sonego P, *et al*. Benchmarking protein classification algorithms via supervised cross-validation. *J Biochem Biophys Methods* 2007 [Epub ahead of print, 31 May 2007].
30. Streiner DL, Cairney J. What's under the ROC? An introduction to receiver operating characteristics curves. *Can J Psychiatry* 2007;**52**:121–8.
31. Ewens WJ, Grant G. *Statistical Methods in Bioinformatics: An Introduction*. 2nd edn, Corr. 2nd printing edn. New York: Springer, 2005.
32. Armitage P, Berry G, Matthews JNS. *Statistical Methods in Medical Research,* 4 edn. Malden, MA: Blackwell Publishing Limited, 2001.
33. Westin LK. *Receiver operating characteristic (ROC) analysis. Evaluating discriminance effects among descision support systems*. Sweden: Umeå universitet - Department of Computing Science, 2001.
34. Dodd LE, Pepe MS. Partial AUC estimation and regression. *Biometrics* 2003;**59**:614–23.
35. Obuchowski NA. Receiver operating characteristic curves and their use in radiology. *Radiology* 2003;**229**:3–8.
36. Jiang Y, Metz CE, Nishikawa RM. A receiver operating characteristic partial area index for highly sensitive diagnostic tests. *Radiology* 1996;**201**:745–50.
37. Salzberg SL. On comparing classifiers: pitfalls to avoid and a recommended approach. *Data Min Knowl Discov* 1997;**1**:317–28.
38. Gribskov M, Robinson NL. Use of receiver operating characteristic (ROC) analysis to evaluate sequence matching. *Comp chem* 1996;**20**:25–33.
39. Schäffer AA, Aravind L, Madden TL, *et al*. Improving the accuracy of PSI-BLAST protein database searches with composition-based statistics and other refinements. *Nucleic Acids Res* 2001;**29**:2994–3005.
40. Jaakkola T, Diekhans M, Haussler D. Using the Fisher kernel method to detect remote protein homologies. *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*, 1999, 149–58.
41. Stephan C, Wesseling S, Schink T, *et al*. Comparison of eight computer programs for receiver-operating characteristic analysis. *Clin Chem* 2003;**49**:433–9.
42. Park SH, Goo JM, Jo C. Receiver operating characteristic (ROC) curve: practical review for radiologists. *Korean J Radiol: Official J Korean Radiol Soc* 2004;**5**:11–8.
43. Web-based Calculator for ROC Curves. www.jrocfit.org (30 September 2007, date last accessed).
44. R: *A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing, 2007.

45. Sing T, Sander O, Beerenwinkel N, *et al*. ROCR. http://rocr.bioinf.mpi-sb.mpg.de/(30 September 2007, date last accessed).

46. Gentleman R, Carey V, Bates D, *et al*. Bioconductor: open software development for computational biology and bioinformatics. *Genome Biol* 2004;**5**:R80.

47. Sonego P, Pacurar M, Dhir S, *et al*. A protein classification benchmark collection for machine learning. *Nucleic Acids Res* 2007;**35**(Suppl 1):D232–6.

48. Frank E, Hall M, Trigg L, *et al*. Data mining in bioinformatics using Weka. *Bioinformatics* 2004;**20**:2479–81.

49. RWeka. http://cran.r-project.org/src/contrib/Descriptions/RWeka.html (30 September 2007, date last accessed).

50. Gewehr JE, Szugat M, Zimmer R. BioWeka—extending the Weka framework for bioinformatics. *Bioinformatics* 2007;**23**:651–3.

51. The receiver operating characteristic (ROC) curve toolkit for MATLAB®. http://theoval.sys.uea.ac.uk/matlab/default.html#roc (30 September 2007, date last accessed).

52. Wootton JC. Evaluating the effectiveness of sequence analysis algorithms using measures of relevant information. *Comp chem* 1997;**21**:191–202.

53. Tax DMJ. *One-class classification; concept-learning in the absence of counter-examples*. Delft, The Netherlands: Delft University of Technology, 2001.

54. Kaján L, Kertész-Farkas A, Franklin D, *et al*. Application of a simple likelihood ratio approximant to protein sequence classification. *Bioinformatics* 2006;**22**:2865–9.

55. Busa-Fekete R, Kertész-Farkas A, Kocsor A, *et al*. Balanced ROC analysis (BAROC) protocol for the evaluation of protein similarities. *J Biochem Biophys Methods* 2007 [Epub ahead of print, 5 July 2007].

56. Noble WS, Kuang R, Leslie C, *et al*. Identifying remote protein homologs by network propagation. *FEBS J* 2005;**272**:5119–28.

57. Weston J, Kuang R, Leslie C, *et al*. Protein ranking by semi-supervised network propagation. *BMC Bioinformatics* 2006;**7**(Suppl 1):S10.

58. Sam V, Tai C, Garnier J, *et al*. ROC and confusion analysis of structure comparison methods identify the main causes of divergence from manual protein classification. *BMC Bioinformatics* 2006;**7**:206.

59. Gibrat JF, Madej T, Bryant SH. Surprising similarities in structure comparison. *Curr Opin Struct Biol* 1996;**6**:377–85.

60. Kocsor A, Kertész-Farkas A, Kaján L, *et al*. Application of compression-based distance measures to protein sequence classification: a methodological study. *Bioinformatics* 2006;**22**:407–12.

61. Leslie CS, Eskin E, Cohen A, *et al*. Mismatch string kernels for discriminative protein classification. *Bioinformatics* 2004;**20**:467–76.