Available online at www.sciencedirect.com
ScienceDirect
J. Biochem. Biophys. Methods 70 (2008) 1215–1223
JOURNAL OF biochemical and biophysical methods
www.elsevier.com/locate/jbbm

Benchmarking protein classification algorithms via supervised cross-validation

Attila Kertész-Farkas [a], Somdutta Dhir [b], Paolo Sonego [b], Mircea Pacurar [b], Sergiu Netoteia [c], Harm Nijveen [d], Arnold Kuzniar [d], Jack A.M. Leunissen [d], András Kocsor [a], Sándor Pongor [b,*]

[a] *Research Group on Artificial Intelligence of the Hungarian Academy of Sciences and University of Szeged, Aradi vértanúk tere 1., H-6720 Szeged, Hungary*
[b] *Protein Structure and Bioinformatics Group, International Centre for Genetic Engineering and Biotechnology, Padriciano 99, I-34012 Trieste*
[c] *Biological Research Centre (BRC), Temesvári krt. 62, Szeged, H-6701 Hungary*
[d] *Laboratory of Bioinformatics, Wageningen University and Research Centre (WUR), Dreijenlaan 3, 6703 HA Wageningen, The Netherlands*

Received 5 April 2007; received in revised form 20 May 2007; accepted 23 May 2007

Abstract

Development and testing of protein classification algorithms are hampered by the fact that the protein universe is characterized by groups vastly different in the number of members, in average protein size, similarity within group, etc. Datasets based on traditional cross-validation (*k*-fold, leave-one-out, etc.) may not give reliable estimates on how an algorithm will generalize to novel, distantly related subtypes of the known protein classes. Supervised cross-validation, i.e., selection of test and train sets according to the known subtypes within a database has been successfully used earlier in conjunction with the SCOP database. Our goal was to extend this principle to other databases and to design standardized benchmark datasets for protein classification. Hierarchical classification trees of protein categories provide a simple and general framework for designing supervised cross-validation strategies for protein classification. Benchmark datasets can be designed at various levels of the concept hierarchy using a simple graph-theoretic distance. A combination of supervised and random sampling was selected to construct reduced size model datasets, suitable for algorithm comparison. Over 3000 new classification tasks were added to our recently established protein classification benchmark collection that currently includes protein sequence (including protein domains and entire proteins), protein structure and reading frame DNA sequence data. We carried out an extensive evaluation based on various machine-learning algorithms such as nearest neighbor, support vector machines, artificial neural networks, random forests and logistic regression, used in conjunction with comparison algorithms, BLAST, Smith-Waterman, Needleman-Wunsch, as well as 3D comparison methods DALI and PRIDE. The resulting datasets provide lower, and in our opinion more realistic estimates of the classifier performance than do random cross-validation schemes. A combination of supervised and random sampling was used to construct model datasets, suitable for algorithm comparison.

The datasets are available at http://hydra.icgeb.trieste.it/benchmark.

Keywords: Protein classification; ROC analysis; Benchmarking

1. Introduction

Classification of protein sequences and protein structures is one of the crucial problems of computational genomics. In a typical application, proteins of a newly sequenced genome are to be classified into one of the several thousand a priori known structural or functional categories, which is carried out by automated methods such as machine-learning algorithms. In view of the large number of new genomes sequenced, it is critically important to define benchmark datasets for assessing the accuracy of

Abbreviations: ROC, receiver operating characteristics; AUC, area under curve, the integral of the ROC curve; BLAST, basic local alignment search tool program of Altschul et al; DALI, distance based structural alignment program of Holm and Sander; PRIDE, protein similarity by probability of fold identity, program of Carugo and Pongor; SCOP, Structural Classification of Proteins, protein domain 3D database of Murzin et al; CATH, protein domain 3D database of Orengo and Thornton.

* Corresponding author.
E-mail addresses: kfa@inf.u-szeged.hu (A. Kertész-Farkas), sdhir@icgeb.org (S. Dhir), sonego@icgeb.org (P. Sonego), pacurar@icgeb.org (M. Pacurar), sergiun@brc.hu (S. Netoteia), harm.nijveenl@wur.nl (H. Nijveen), arnold.kuzniar@wur.nl (A. Kuzniar), jack.leunissen@wur.nl (J. Leunissen), kocsor@inf.u-szeged.hu (A. Kocsor), pongor@icgeb.org (S. Pongor).

0165-022X/$ - see front matter © 2007 Published by Elsevier B.V.
doi:10.1016/j.jbbm.2007.05.011

classification algorithms. This work is undertaken in order to propose a general framework in which benchmark datasets can be constructed.

In the practice of machine learning, cross-validation techniques are used to assess the accuracy of classification methods. Cross-validation is based on training a machine-learning algorithm on one, randomly picked part of the data and then testing it on another part (this is the so-called holdout method). In the k-fold cross-validation approach the data are randomly divided into k disjunct subsets approximately equal in size, and the holdout method is repeated k times, using one of the k subsets as the test set while the other $k-1$ subsets are put together to form a training set. Subsequently the average of a performance measure across all k trials is computed. Finally, the leave-one-out approach is a k-fold technique with $k=1$, i.e., each object is considered a test set in itself while the training is carried out on the rest of the database.

The principle underlying cross-validation is the attempt to produce training sets that best approximate the distribution of the expected test samples, and this is achieved with randomly picked subsets. Protein classification however requires a different approach, because of the biological nature of the problem itself. Namely, most genomes contain novel variants of the known proteins, i.e., the similarity distribution of a known protein family in a newly sequenced genome is in fact expected to be different from rather than similar to that of its known variants. So the property relevant to biologists is how well a classifier will generalize to novel types of the already known groups. The random cross-validation methods mentioned earlier do not provide information on this property. However, one can assess the generalization capability of a classifier using additional knowledge on the protein classes. For example, if it is known that a given group consists of subgroups, one can use one subgroup as the positive test set and pool the others as the positive training set. One can repeat this procedure for each of the subgroups. With this method each of the subgroups is considered one-by-one as a "newly discovered" subtype, so the method will estimate the classifier's average ability to discover new variants. This technique is similar to k-fold cross-validation; however, the subgroups are based on our knowledge, so we can speak about a knowledge-based or supervised cross-validation strategy as opposed to the random cross-validation techniques. Such a subdivision of the data was applied previously to SCOP superfamilies and families [2–6].

The aim of the present work is to explore the principles of developing benchmark datasets for protein classification based on arbitrary–hierarchical schemes. Our starting point is that the generalization capabilities of an algorithm can be characterized by testing how well it is able to recognize the already known subgroups within a given group. Since the seminal paper of Lindahl and Elofsson [6] these studies were restricted mostly to the superfamily/family level of the SCOP database [3–5,7], here we show that the process can be generalized to any level of the structural hierarchy, and in fact to any database on which a hierarchical classification scheme is defined, such as CATH [7], COG [8] or UNIPROT [9].

Recently we established a collection of protein classification tasks [1]. With the present work we seek to lay down the general rules how to construct benchmark datasets, and to answer few specific questions: (i) do supervised and random cross-validation schemes provide different results (Sections 3.1 and 3.2)? (ii) How critical is the choice of the negative sets, and how can we restrict their size to computationally manageable numbers (Sections 3.3). (iii) Can we consistently rank comparison measures and machine-learning algorithms based on the classification experiments, i.e., is it meaningful to design small benchmark datasets for algorithm development (Sections 3.4)? Sections 4 summarizes the conclusions and Sections 5 gives a brief summary description of the method.

2. Materials and methods

2.1. Datasets

SCOP95. The sequences were taken from the SCOP database 1.69 [10]. The entries of the SCOP95 (<95% identity) were downloaded from the ASTRAL site http://astral.berkeley.edu/. 121 Non-contiguous domains were discarded and 11944

Table 1
Comparison of random and supervised cross-validation strategies on the example of the Lipocalin superfamily [a] of the SCOP database

	A		Family B	No. of family members in +test C	No. of family members in +train D	Area under curve E	Average area under curve F
(1)	Leave one out		Retinol-binding	1	27	0.9908	0.9908
			Fatty acid binding	1	26		
(2)	5-Fold cross-validation		Retinol-binding	3–7	19–20	0.9996, 0.9982, 1.0000, 0.9979,	0.999127 (0.99–1.0)
			Fatty acid binding	4–7	19–23	1.0000	
		1	Retinol-binding	28	0	0.8635	
(3)	Supervised cross-validation		Fatty acid binding	0	27		0.8243 (0.79–0.86)
		2	Retinol-binding	0	28	0.7851	
			Fatty acid binding	27	0		

[a] The superfamily consists of 58 sequences, 27 in the fatty acid binding family and 28 in the retinol-binding family, as well as 3 further sequences that, in the supervised case, were not used as members of the +test group. The negative group was the rest of the SCOP95 database which was randomly subdivided either into equal test and train groups [in cases (1) and (3)] or into 80% −train and 20% −test groups (2). The AUC value characterizes the efficiency of the ranking, for perfect ranking it is 1.00, for random ranking it is around 0.5.

Table 2
Classification performance calculated on the SCOP95 dataset in which the negative set was subdivided in various ways

Division of −set into test and train	1NN	SVM
A	B	C
By sequence	0.8397±0.1696	0.8698±0.1408
By family	0.8352±0.1743	0.8641±0.1472
By superfamily	0.8435±0.1654	0.8686±0.1418
By fold	0.8193±0.1849	0.8455±0.1628

Sequence comparison by the Smith-Waterman algorithm. Results are average± standard deviation of 246 classification tasks in each of which a SCOP family was the +test set for the respective superfamily (total no of sequences=11 944). The negative set was the rest of the database (11,000–11,500 sequences) which were divided in various ways as indicated in column A. Subdivision by sequence corresponds to random division, the others rely on the knowledge of the subgroups.

entries were retained. The sequences were stored in concatenated FASTA format. The subdivisions into +test, +train, −test, −train sets are described in Tables 1 and 2.

CATH95. The sequences were taken from the CATH database v.3.0.0 [7]. The entries of the CATH95 (>95% identity) selection were downloaded from the ftp://ftp.biochem.ucl.ac.uk/pub/cathdata/v3.0.0/ site. The 1648 non-contiguous domains were discarded and 11373 were retained. The sequences were stored in concatenated FASTA format.

SCOP40mini. The sequences were taken from the SCOP database 1.69 [10]. The entries of the SCOP40 (<40% identity) were downloaded from the ASTRAL site http://astral.berkeley.edu/. 53 non-contiguous domains were discarded and 7237 entries were retained. The sequences were stored in concatenated FASTA format. Protein families, with at least 5 members and at least 10 members outside the family but within the same superfamily in SCOP95 were selected. From this selection those families were selected as positive test that had at least 10 members outside the family but within the same superfamily in the selection (total: 1357 proteins/55 classification tasks).

The subdivision of the various datasets into +test, +train, −test, −train sets are described in Tables 3, 4, 5, 6 and 7. For further details, see http://hydra.icgeb.trieste.it/benchmark.

2.2. Sequence comparison methods

The sequence comparisons were made with the following programs: (i) BLAST version 2.2.13 of the BLAST program [11] using a cutoff score of 25; (ii) the Smith-Waterman algorithm as implemented in MATLAB [12]; (iii) the Needleman-Wunsch algorithm [13] as implemented in the needle program of EMBOSS [14], using a gap opening penalty of 10 and a gap extension penalty of 0.5. (In i–iii the BLOSUM 62 matrix [15] was used.); (iv) the Local Alignment Kernel program version 0.3 of Saigo and associates [16] was downloaded from http://cg.ensmp.fr/~vert/. The following run parameters were used: default comparison matrix found in the parameters.h file. Gap opening penalty=11 (default), Gap extension penalty=1 (default), Scaling parameter=0.5.

Comparisons with compression distance functions were calculated as described by Kocsor and associates [17], using the LZW [18] or the PPMZ algorithms [19]. The LZW algorithm was implemented in C, while the PPMZ2 algorithm was downloaded from Charles Bloom's homepage (http://www.cbloom.com/src/ppmz.html).

2.3. Classifier algorithms

Nearest neighbor (1NN) classification [20] is a simple technique whereby a sequence is assigned to the a priori known class of the database entry that was found most similar to it in terms of a distance/similarity measure.

Support vector machines (SVMs) were used as described in [5]. Sequence X was represented by a feature vector $F_X=f_{x1}, f_{x2}, \ldots f_{xn}$, where n is the total number of proteins in the training set and f_{xi} is a similarity/distance score, such as the BLAST score, the Smith–Waterman score or a compression-based distance between X and the ith sequence in the training set. For the SVM experiments we used the SVMlight software package [21] as described by Liao and Noble [5].

The Logistic Regression (LogReg) is a variation of ordinary regression which is used when the response variable is dichotomous (i.e., it takes only two values, usually coded as −1 and 1) and the input variable is continuous, categorical or both. The LogReg algorithm was used as implemented in the WEKA software package ([22], Version 3.4.7, December 2005, downloaded from http://www.cs.waikato.ac.nz/ml/weka/).

The Random Forest (RF) technique uses a combination of decision trees [23]. The script uses 10 trees and the number of variables in each node is set to $\log(D+1)$, where D is the number of inputs. The script uses the Random Forest library of *R*, written by Andy Liaw and Matthew Wiener based on Breiman's original code. (http://cran.r-project.org/src/contrib/PACKAGES.html).

2.4. Evaluation of classifier performance

The evaluation was carried out by the standard receiver operating characteristic (ROC) analysis [24]. This method is especially useful for protein classification as it includes both sensitivity and specificity, based on a ranking of the objects to be classified [25]. In the case of 1NN classifications, the ranking variable was the similarity or distance value calculated between a query sequence and its nearest neighbor in the positive training set. This scenario corresponds to one-class classification with outlier detection. Briefly, the analysis is carried out by plotting

Table 3
Classification of SCOP95 sequences and structures used in this study

SCOP95 classes	#Sequences	#Families	#Superfamilies	#Folds
Total	11944	2834	1532	941
α	2141	607	375	218
β	3077	559	289	143
α/β	2801	629	222	136
α+β	2612	711	407	278
Multidomain	204	60	45	45
Membrane and cell surface	222	98	87	47
Small	887	170	107	74

Table 4
Classification of SCOP40 sequences and structures used in this study

SCOP40 classes	#Sequences	#Families	#Superfamilies	#Folds	#+Test sequences	#+Test families used
Total	1357	291	24	22	771	55
α	258	102	5	4	85	8
β	377	65	6	5	226	15
α/β	679	113	11	11	445	30
α+β	23	5	1	1	10	1
Multidomain	20	6	1	1	5	1
Membrane and cell surface	0	0	0	0	0	0
Small	0	0	0	0	0	0

sensitivity vs. 1-specificity at various threshold values, then the resulting curve was integrated to give an "area under curve" or AUC value. If the evaluation procedure contains several ROC experiments, one can draw a cumulative distribution curve of the AUC values [5], or use the average of the individual AUC values as performance indicator.

3. Results

A system capable of testing and comparing machine-learning algorithms includes (i) datasets and classification tasks; (ii) sequence/structure comparison methods; (iii) the classification algorithms themselves, and (iv) a validation protocol. This work is centered on the design of datasets and classification tasks, the rest of the ingredients, including the datasets (protein sequences, protein structures, protein coding DNA sequences) and machine-learning algorithms (nearest neighbor analysis, support vector machines, artificial neural network, random forests and logistic regression) are described in the Materials and methods section.

3.1. Random sampling vs. supervised classification: an example

In order to train a classifier algorithm, we need a database subdivided into positive and negative groups. In the next step we have to subdivide these two groups into test and train sets. The result is a subdivision of the dataset into +train, +test, −train and −test groups that will be used for training and testing a classifier algorithm. We will term this fourfold subdivision a classification task. In order to test a machine-learning method on an entire database, we need to design classification tasks for all possible classes of the dataset. We will term the ensemble of the classification tasks a benchmark test.

Let us take first a classification task that illustrates the difference between random and knowledge-based (supervised) subdivision of the samples: the Lipocalin superfamily of the SCOP95 database contains 58 sequences. These will be the positive set, while the rest of the database will be the negative set. The evaluation will be based on nearest neighbor classification, based on a Smith-Waterman comparison. If we follow the traditional strategy of machine learning, we can subdivide this set randomly. For example, we can use each of the members as +test and use the rest of the superfamily as positive train (leave-one-out strategy). Or we can use, for instance a 5-fold cross-validation strategy where a randomly chosen one fifth of the group is used as +test, and the remaining four fifth will be +train. The negative set will subdivided in an analogous way. One evaluation will consist of ROC/AUC calculation (in which AUC=1 indicates perfect ranking of the samples in terms of the Smith-Waterman score, AUC=0.5 corresponds to random ranking). As shown in Table 1, both of these tests give rather high results, the AUC values are close to 1.00. However, if we divide the samples in a supervised manner, i.e., according to the known subgroups (lines 3–4 of the table), then we get substantially lower AUC values. The results can be better understood if we look at the number of the subgroup members included in the +test and +train groups (columns D and E, respectively). In the two random subdivisions, the retinol-binding and the fatty-acid-binding subgroups are included in both the +test and the +train sets, so the comparison scores will be high. In the supervised subdivisions, on the other hand, members of the subgroups make part either of the +test or of the +train group, so the comparison scores will be lower, resulting in lower AUC values. The two supervised calculations thus refer to situations

Table 5
Classification of CATH95 sequences and structures used in this study

CATH95 classes	#Sequences	#H groups	#T groups	#A groups
Total	11373	1960	989	39
α	2672	628	279	5
β	3334	393	176	19
α−β	5107	839	445	14
fewSS	260	100	89	1

Table 6
Distribution of proteins in benchmark tests defined on SCOP95 dataset

SCOP95	(1) Superfamilies divided by families		(2) Folds divided by superfamilies		(3) Classes divided by folds	
	# +Test sequences	# +Test families used	# +Test sequences	# +Test families used	# +Test sequences	# +Test families used
Total	4323	246	3901	650	10927	2187
α	614	43	522	145	1899	453
β	1507	55	1727	178	2921	466
α/β	1392	86	734	150	2675	557
α+β	503	41	583	134	2300	505
Multidomain	33	4	0	0	148	24
Membrane and cell surface	0	0	27	5	167	62
Small	274	17	308	38	817	120

Table 7
Distribution of proteins in benchmark tests defined on CATH95 dataset

CATH95	(1) Homology groups divided into sequence similarity groups		(2) Topology groups divided by homology groups		(3) Architecture divided by topology groups		(4) Classes divided by architecture groups	
	# +Test sequences	# +Test H-groups used	# +Test sequences	# +Test H-groups used	# +Test sequences	# +Test H-groups used	# +Test sequences	# +Test H-groups used
Total	1832	649	5288	1310	9938	1846	11103	1939
α	503	198	1277	403	2329	594	2590	617
β	498	134	1508	262	2896	360	3253	390
α−β	773	282	2370	578	4478	800	5009	833
fewSS	58	35	133	67	235	92	251	99

where we attempt to predict one subgroup based on the other one, i.e., they estimate the generalization capability of a method on this particular classification task.

3.2. Generalization to arbitrary classification hierarchies

Let us consider a database consisting of N (sequence or structure) objects O_1, O_2, … O_N with a proximity measure $M(O_i, O_j)$ defined between the objects. A proximity measure is a common term for similarity measures (e.g., sequence similarity scores, association measures for feature vectors) and dissimilarity measures (e.g., Euclidean distances of feature vectors). The neighborhood $K(O_i, d_1, d_2)$ of object O_i is defined as a subset of the other, remaining O_j objects for which d_1 $M(O_i, O_j)d_2$, where d_1 and d_2 are the limits of the object's neighborhood. If $d_1=d_2=d$ we can simply write $K(O_i, d)$. The object neighborhood is thus a partitioning of the database into neighbors and non-neighbors with respect to an object O_i. We can describe the neighborhood of a group A in a similar fashion, provided we can define a proximity measure between group A and the objects O_j outside the group. Since function $M(O_i, O_j)$ is either a similarity or a dissimilarity measure, we can simply define $M(A, O_j)$ for any O_j either as the minimal distance or the maximal similarity between O_i and any member of A. The group neighborhood is thus a partitioning of the database into neighbors and non-neighbors with respect to a group A. Following the above notational convention we denote the neighborhood of a particular group A by $K(A, d_1, d_2)$ and $K(A, d)$.

Let us assume that a database consist of objects that are defined according to terms arranged into a hierarchical classification tree. Then the dataset can be regarded as a rooted tree that is a simple, undirected, connected, acyclic graph. Moreover, it has one special labeled node called a root, and if a node has only one edge it is called a leaf. Denote $D(e, f)$ the distance between node e and node f, which is defined here as the number of edges on the shortest path between e and f. The node distance of a node from the root is called the depth of the node. We define the set of nodes at depth i – or level i, denoted by $L(i)$ – as consisting of nodes having the same depth value. Formally, where e is a node and root is the root node of the tree. We call a tree a balanced tree, if all the depth of the leaves are the same, and this distance is called the height of the tree, denoted by H. From now on when we mention a tree, we always mean a balanced tree. Fig. 1 shows a typical example of a balanced tree in which the leaves are the protein entries of the database and the root of the tree is the database itself. Each of the other nodes defines a subgroup of protein entries that are the leaves connected to the given node. In this way the nodes in $L(i)$ represent a partition of the database into disjoint groups labeled by the categories at level i.

Using the tree hierarchy we can define group neighborhoods by applying the $D(e, f)$ distance as a proximity measure over proteins (i.e., the leaves), which provides a supervised way to partition the database.

In order to construct supervised classification tasks for a given database, we need to know the subdivision at least two adjacent hierarchical levels (e.g., superfamily/family). The + and − groups are defined at the higher level (i), while the learning/test subdivision is at the lower level ($i+1$).The principle is highlighted in Fig. 1A, where we use the number of steps between two proteins within the hierarchy, to define various subgroups. In the figure, the positive sets are members of the same family, which are 2 steps away from each other. The members of the negative set are on the other hand 4 steps away from any member of the positive set. Because of its generality, this principle can be applied both to other levels of this hierarchy as well as to other tree hierarchies. The statistical description of a few such datasets is described in

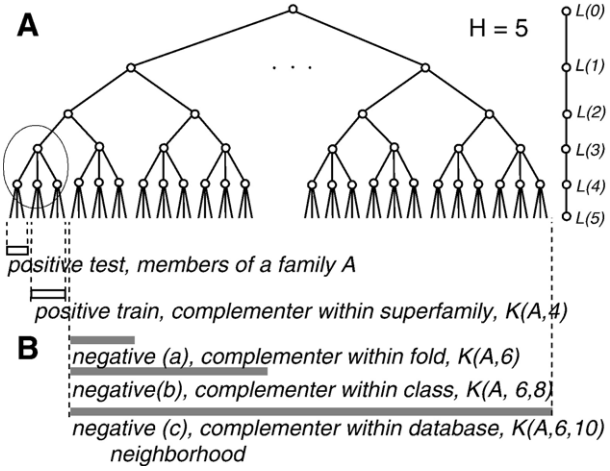

Fig. 1. Application of supervised cross-validation scheme to an arbitrary classification hierarchy. (A) Definition possibilities for positive and negative sets within a classification hierarchy. The hierarchy is a schematic and partial representation of that of the SCOP database [9]. The positive set is defined at the superfamily level, the +test and +train sets are defined at the underlying family level. (B) The boundaries of the negative set can be fixed in terms of the number of steps within the tree hierarchy, calculated with respect to the positive set A. For instance, $K(A, 4)$ defines a neighborhood (a) whose members are 4 steps apart from the members of group A. For detailed explanations see Materials and methods.

Tables 3, 4 (deposited as Supplementary information at the database site).

Fig. 2 compares supervised and the random cross-validation methods at various levels of the SCOP and CATH hierarchies. The results show that the AUC values of random cross-validation tests are substantially higher than the supervised values, i.e., the qualitative picture obtained on the example shown in Table 1 is in fact general to all the classification levels of SCOP and CATH. At all levels of the hierarchies there is a clear tendency in the ranking: leave-one-out test ~5-fold cross-validation > knowledge based cross-validation. This tendency indirectly explains why an excellent performance obtained with random cross-validation techniques does not necessarily guarantee good performance on sequences from new genomes, in other words random cross-validation techniques may grossly overestimate the predictive power of a method on new genomes.

The tendencies shown in Fig. 2 confirm the well-known fact that the prediction at the lower levels of the hierarchy is more efficient than at the higher levels [1]. It is also apparent that the difference between the random and the supervised subdivision is larger at the higher levels of the hierarchy, in spite of the fact that the domain definitions and the hierarchies of SCOP and CATH are different. On the other hand, the differences of the two databases are reflected by the different shapes of the corresponding curves.

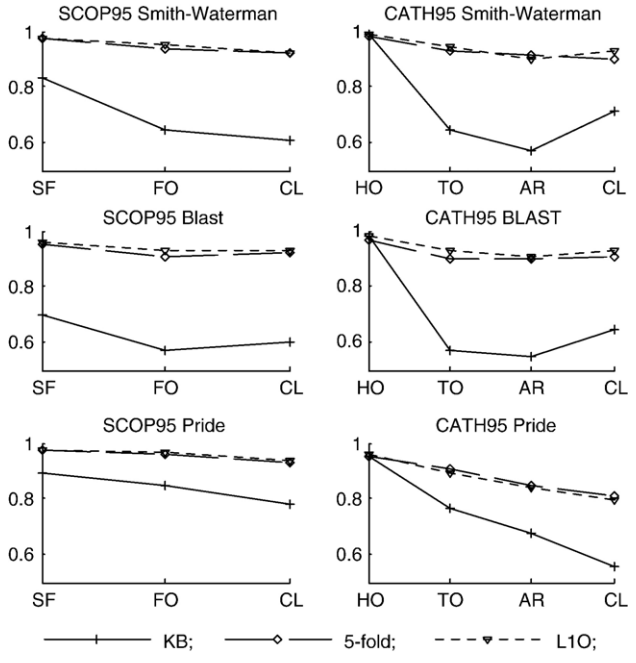

Fig. 2. The comparison of supervised and random cross-validation schemes on the SCOP and CATH databases benchmark tests on the SCOP95 (left) and CATH 95 (right) datasets conducted at various levels of the classification hierarchy, using Smith-Waterman (top), BLAST (middle) for sequence comparison and PRIDE (bottom) for structure comparison. Categories on the *X* axis are the levels of the classification (In SCOP: SF: supefamilies; FO: folds divided; CL; classes; In CATH: HO: homology groups; TO: topology groups; AR; architecture groups; CL: classes), the *Y* axis shows the average AUC in a benchmark test. KB = supervised (knowledge-based) (+); L1O = (◇) leave-one-out, *5*-fold cross-validation (▽). Note that the random subdivisions give higher results than the supervised (knowledge-based) ones.

In the above calculations, we varied only the subdivision of the positive set and kept the subdivision of the negative set constant. Naturally one can also vary the way how the negative test is subdivided (Table 2). For instance, subdivision by superfamily means that members of a superfamily can be either −train or −test. This subdivision corresponds to a hypothetical situation where a newly sequenced genome contains only novel superfamilies that have not been used for training. This is a stringent test, since subdivision at the fold level would mean that the new genome contains only novel folds, which is not a likely event. Subdivision by sequence, on the other hand corresponds to the random subdivision strategy employed in the practice of machine learning. Table 2 shows that the classification performance is not very sensitive to this choice, even though a slight decrease of performance is seen as we increase the stringency of the test.

3.3. Selecting negative datasets of manageable sizes

In a typical classification task described in the previous sections, the positive sets consisted of 15 to a few hundred sequence objects but the negative set was simply defined as “the rest of the database”. In other words, the dataset was large and imbalanced because of the size of the negative sets. Filtering the negative set is a plausible idea and we compare three strategies for doing this: (i) random subsampling of the negative set (10, 20 or 30%); (ii) selection of a nearest neighbors of the positive group [26], based on a Smith-Waterman score; and (iii) selection of the category neighbors. The latter is a supervised selection, and the principle is shown in Fig. 1B. If the positive set is a given superfamily, the negative set can be selected as other superfamilies within the same fold (a), or other superfamilies within the same class (b).

In the calculations summarized in Table 8, classification performance was characterized by the average and the standard deviation of 246 classification tasks defined on the SCOP database. A higher performance is characterized by a higher average and a lower standard deviation. The results show that random subsetting of the negative group does not substantially influence the classification performance either of the 1NN or of the SVM classifiers. However, when we choose the nearest neighbors according to the Smith-Waterman algorithm, there is a decrease in the performance of the 1NN classifier but the performance of the SVM classifiers remains roughly the same. On the other hand, when we select the negative sets on the basis of the classification hierarchy, both classifiers show a performance decrease. The number of structural categories present in the negative groups provides some insight into these tendencies.

Selecting the nearest neighbors in terms of a Smith-Waterman distance or in terms of structural hierarchy sharply reduces the number of structural categories present in the negative set. The second method also produces very small datasets. As a result of these two factors, the negative sets will be too specific, i.e., they may be less representative with respect to the entire database. Based on the above results, random filtering of the negative sets seems to be the most sensible

Table 8
Dependence of classifier performance and the subgroup composition on the selection of the negative set (SCOP95 dataset, Smith-Waterman score)[a]

Size of negative set (%)	1NN	SVM	Number[b] of			
	AUC±SD[b]		Families	Superfamilies	Folds	Classes
Full negative set (11162–11929 sequences)[c]						
100	0.8352±0.17	0.8641±0.15	2824 (2779–2832)	1531 (1531–1531)	941 (940–941)	7
Random selection[d]						
10	0.8352±0.13	0.8865±0.17	734 (700–764)	520 (480–552)	369 (341–395)	7
20	0.8351±0.14	0.8776±0.17	1182 (1124–1223)	770 (728–810)	514 (480–553)	7
30	0.8353±0.17	0.8754±0.14	1522 (1458–1572)	943 (902–983)	609 (577–637)	7
Nearest neighbor selection[e]						
10	0.5836±0.32	0.8738±0.19	452 (383–559)	270 (226–347)	205 (164–265)	6 or 7
20	0.6410±0.23	0.8954±0.16	810 (727–940)	438 (384–536)	328 (280–403)	6 or 7
30	0.7389±0.28	0.8880±0.17	1136 (1038–1287)	595 (529–717)	434 (390–510)	6 or 7
Supervised selection[f] *(knowledge based)*						
A	0.9±1.3	0.7083±0.21	0.7651±0.19	26 (2–79)	11 (1–30)	1
B	20.8±4.8	0.7571±0.18	0.8664±0.14	558 (165–705)	287 (106–406)	1

[a] AUC values were calculated for the 246 classification tasks defined on the SCOP95 dataset. The results were calculated as the average and standard deviation (SD) of the AUC values. The random selection was repeated 10 times, and the SD value is the average of the 10 calculations.
[b] The numbers indicate average (minimum–maximum) within the classifier tasks.
[c] The negative sets (11,162–11,929 sequences) were subdivided into roughly equal −train and −test groups in such a way that members of a given family were either train or test.
[d] The given percentage of the −train and −test set was selected randomly.
[e] The members of the −test and −train sets were ranked according to their highest Smith-Waterman score with respect to the +train group, and the closest sequences corresponding to the given percentage were selected.
[f] The selection was based on the ranges indicated by panels (a) and (b) in Fig. 1.

compromise, since the classification performances remain close to those of the original dataset, and the number of structural categories is higher than in the case of the other two methods.

Finally we mention that the difference between the behavior of 1NN and SVM in these calculations may be due to the fact that the ROC analysis of the 1NN is based on a one-class scenario (outlier detection), whereas the performance of SVMs is evaluated with respect to a decision surface separating the +train and −train classes.

3.4. Comparison of learning algorithms using various similarity/dissimilarity measures

We selected a number of popular similarity/distance measures and learning algorithms for the comparisons summarized in Tables 9–11. We used datasets selected so as to represent different types of classification problems. The 3PGK dataset consists of the sequences of the ubiquitous enzyme 3-phosphoglycerate kinase, grouped into superkingdoms (Archaea, Bacteria and Eukaryota), each subdivided into phyla as described in [27,28]. In this dataset the sequence similarities within the groups (i.e., between members of the same superkingdom) are relatively high and between the groups (between members of different superkingdoms) are also relatively high. The situation is different in the SCOP95 database, where both the within-group and the between-group sequence similarities are relatively low. Finally the SCOP40 dataset is even more difficult since here sequences more similar to each other than 40% are represented by a single prototype sequence.

Tables 9–11 summarize the classification performance data obtained with the various methods (columns) and various sequence/structure comparison algorithms (rows). We mention that because of time constraints some of the comparison algorithms (PPMZ, DALI) were not calculated for the biggest

Table 9
SCOP40 mini-dataset (superfamilies, subdivided into families): comparison of distance measures and machine learning algorithms

	1NN	RF	SVM	ANN	LogReg
BLAST	0.7577	0.6965	0.9047	0.7988	0.8715
SW	0.8154	0.8230	0.9419	0.8875	0.9063
NW	0.8252	0.8030	0.9376	0.8834	0.9175
LA	0.7343	0.8344	0.9396	0.9022	0.8766
LZW	0.7174	0.7396	0.8288	0.8346	0.7487
PPMZ	0.5644	0.7253	0.8551	0.8254	0.8308
PRIDE	0.8644	0.9105	0.9361	0.9073	0.9029
DALI	0.9892	0.9941	0.9946	0.9897	0.9636

Table 10
SCOP95-10 dataset (superfamilies, subdivided into families): comparison of distance measures and machine learning algorithms

	1NN	RF	SVM	ANN	LogReg
BLAST	0.6985	0.6729	0.7293	0.5703	0.7218
SW	0.8354	0.8645	0.8884	0.8607	0.8574
NW	0.8390	0.8576	0.8910	0.8393	0.8607
LA	0.7884	0.8823	0.8717	0.9002	0.8718
LZW	0.7830	0.8208	0.8402	0.7851	0.7574

Table 11
3PGK dataset (superkingdoms divided into phyla): comparison of distance measures and machine learning algorithms

	1NN	RF	SVM	ANN	LogReg
(A) Protein sequences					
BLAST	0.8633	0.8517	0.9533	0.9584	0.9537
SW	0.8605	0.8659	0.9527	0.9548	0.9476
NW	0.8621	0.8548	0.9542	0.9547	0.9494
LA	0.8596	0.8755	0.9549	0.9564	0.9593
LZW	0.7833	0.8463	0.9242	0.9278	0.9154
PPMZ	0.8117	0.9152	0.9476	0.9597	0.9398
(B) DNA sequences					
BLAST	0.7358	0.8244	0.8102	0.8077	0.7691
SW	0.8864	0.8674	0.9630	0.9698	0.9772
NW	0.8437	0.8959	0.9455	0.9433	0.9612
LA	n/a	n/a	n/a	n/a	n/a
LZW	0.7143	0.7107	0.8343	0.8297	0.7844
PPMZ	0.7336	0.7662	0.7881	0.7918	0.8612

datasets. The results show a sufficiently clear tendency: the ranking of the machine-learning methods is roughly the same on all datasets, i.e., usually SVM gave the best performance closely followed by ANN and LogReg. We mention that this ranking does not refer to the potentials or the optimum performance of the machine-learning algorithms, it rather reflects their actual performance obtained with the default settings employed here.

On the other hand, the ranking of the similarity–dissimilarity measures shows some variation between the datasets. In general we see the expected trend, i.e., the 3D comparison algorithms are better that sequence-based methods, and exhaustive methods (Smith-Waterman, Needleman-Wunsch) perform better than heuristic algorithms (BLAST). On the other hand, BLAST performs slightly better than Smith-Waterman on the 3PGK DNA sequences. These variations are in a way expected, knowing that the within-group and between-group similarities are different in the various datasets.

4. Discussion

Supervised cross-validation is a strategy that allows one to estimate the capability of an algorithm to recognize novel subtypes of known categories. As novel protein types abound in newly sequenced genomes, generalization capability of an algorithm is crucial for genome annotation.

One can design classification tasks in a supervised way if there are known subclasses within the classes to be studied. If the categories are hierarchically organized, one can define classification tasks at various levels of the hierarchy. Here we described a number of such classification tasks that vary in the degree of difficulty and characterized them using a number of comparison methods (e.g., BLAST, Smith-Waterman, Needleman-Wunsch, as well as 3D comparison methods) as well as classifier algorithms (nearest neighbor, support vector machines, artificial neural networks, etc.). Even though the results vary from dataset to dataset, the ranking on machine-learning algorithms is grossly consistent on all the datasets. On the other hand, the ranking of the comparison algorithms appears to be more dataset-dependent, and there is a large variation between the individual groups of the same dataset. So the final evaluation of an algorithm should be made on the groups of real-life datasets.

The protein data (sequences, 3D structures, reading-frame DNA sequences) as well as several precomputed similarity/distance matrices are now deposited into a recently established, publicly available data collection that also includes evaluation results (ROC/AUC values) for several classifier algorithms [1]. Based on the results of the present works we added to the collection 14 new, reduced size datasets (with a total of 3042 new classification tasks) that may facilitate the use by the pattern matching community. We hope that these datasets will be useful for method developers as well as for users interested in assessing the performance of various methods.

5. Simplified description of the method

We present a method for testing the generalization capability of protein classification algorithms. It consists in building a set of classification tasks (+train, +test, −train, −test groups) for a protein database that has a category hierarchy (such as protein domains databases, protein family databases, phylogenetic hierarchies etc.). The subdivision – termed *supervised cross-validation* – is based on two successive classification levels, the + and − groups are defined at the higher level (i), while the learning/test subdivision is at the lower level ($i+1$). Datasets of manageable sizes suitable for testing/developing machine-learning algorithms can then be designed by randomly selecting a smaller (say 10 or 20%) subset from the negative sets.

Randomly subdivided test datasets routinely used by the machine-learning community give erroneously good results for protein classification which overestimate predictor performance on new genomic data. The method described here allows one to get a realistic estimate regarding a predictor's performance with respect to novel data and is applicable essentially to any datasets wherein the objects are classified in a hierarchical manner.

Acknowledgements

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] Sonego M, Pacurar S, Dhir A, Kertesz-Farkas A, Kocsor Z, Gaspari JA, et al. A protein classification benchmark collection for machine learning. Nucleic Acids Res 2007;35:D232–6.
[2] Dong QW, Wang XL, Lin L. Application of latent semantic analysis to protein remote homology detection. Bioinformatics 2006;22:285–90.
[3] Jaakkola T, Diekhans M, Haussler D. Using the Fisher Kernel method to detect remote protein homologies. Proc Int Conf Intell Syst Mol Biol 1999:149–58.
[4] Jaakkola T, Diekhans M, Haussler D. A discriminative framework for detecting remote protein homologies. J Comput Biol 2000;7:95–114.
[5] Liao L, Noble WS. Combining pairwise sequence similarity and support vector machines for detecting remote protein evolutionary and structural relationships. J Comput Biol 2003;10:857–68.

[6] Lindahl E, Elofsson A. Identification of related proteins on family, superfamily and fold level. J Mol Biol 2000;295:613–25.
[7] Pearl F, Todd A, Sillitoe I, Dibley M, Redfern O, Lewis T, et al. The CATH Domain Structure Database and related resources Gene3D and DHS provide comprehensive domain family information for genome analysis. Nucleic Acids Res 2005;33:D247–51.
[8] Tatusov RL, Fedorova ND, Jackson JD, Jacobs AR, Kiryutin B, Koonin EV, et al. The COG database: an updated version includes eukaryotes. BMC Bioinformatics 2003;4:41.
[9] Wu CH, Apweiler R, Bairoch A, Natale DA, Barker WC, Boeckmann B, et al. The Universal Protein Resource (UniProt): an expanding universe of protein information. Nucleic Acids Res 2006;34:D187–91.
[10] Andreeva A, Howorth D, Brenner SE, Hubbard TJ, Chothia C, Murzin AG. SCOP database in 2004: refinements integrate structure and sequence family data. Nucleic Acids Res 2004;32:D226–9.
[11] Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. Basic local alignment search tool. J Mol Biol 1990;215:403–10.
[12] MathWorks T. Matlab. The MathWorks. MA: Natick; 2004.
[13] Needleman SB, Wunsch CD. A general method applicable to the search for similarities in the amino acid sequence of two proteins. J Mol Biol 1970;48:443–53.
[14] Rice P, Longden I, Bleasby A. EMBOSS: The European Molecular Biology Open Software Suite. Trends Genet 2000;16:276–7.
[15] Henikoff S, Henikoff JG, Pietrokovski S. Blocks+: a non-redundant database of protein alignment blocks derived from multiple compilations. Bioinformatics 1999;15:471–9.
[16] Saigo H, Vert JP, Ueda N, Akutsu T. Protein homology detection using string alignment Kernels. Bioinformatics 2004;20:1682–9.
[17] Kocsor A, Kertesz-Farkas A, Kajan L, Pongor S. Application of compression-based distance measures to protein sequence classification: a methodological study. Bioinformatics 2006;22:407–12.
[18] Lempel A, Ziv J. On the complexity of finite sequences. IEEE Trans Inform Tehory 1976;22:75–81.
[19] Bloom C. Solving the problems of context modelling. California Institute of Technology 1998:1–11.
[20] Duda RO, Hart PE, Stork DG. Pattern classification. New York: John Wiley & Sons; 2000.
[21] Joachims T. Making large-scale support vector machine learning practical. In: Schoelkopf CCB, Smola A, editors. Advances in Kernel methods: support vector machines. Cambridge, MA: MIT Press; 1998.
[22] Witten IH, Frank E. Data mining: practical machine learning tools and techniques. Second Edition. San Francisco, CA: Morgan Kaufmann; 2005.
[23] Breiman L. Random forests. Mach Learn 2001;45:5–32.
[24] Egan JP. Signal detection theory and ROC analysis. New York; 1975.
[25] Gribskov M, Robinson NL. Use of receiver operating characteristic (ROC) analysis to evaluate sequence matching. Comput Chem 1996;20:25–33.
[26] Murvai J, Vlahovicek K, Szepesvari C, Pongor S. Prediction of protein functional domains from sequences using artificial neural networks. Genome Res 2001;11:1410–7.
[27] Kajan L, Kertesz-Farkas A, Franklin D, Ivanova N, Kocsor A, Pongor S. Application of a simple likelihood ratio approximant to protein sequence classification. Bioinformatics 2006;22:2865–9.
[28] Pollack JD, Li Q, Pearl DK. Taxonomic utility of a phylogenetic analysis of phosphoglycerate kinase proteins of Archaea, Bacteria, and Eukaryota: insights by Bayesian analyses. Mol Phylogenet Evol 2005;35:420–30.