Appears in *Proc. of the 3rd International Conference on Intelligent Systems for Molecular Biology*,
C. Rawlings, D. Clark, R. Altman, L. Hunter, T. Lengauer & S. Wodak (eds.), AAAI Press, Menlo Park, CA, 1995.

Predicting Protein Folding Classes without Overly Relying on Homology*

Mark W. Craven
Computer Sciences Department
University of Wisconsin-Madison
1210 W. Dayton St.
Madison WI 53706
craven@cs.wisc.edu

Richard J. Mural
Biology Division
Oak Ridge National Laboratory
P.O. Box 2008, Bldg. 9211
Oak Ridge TN 37831-8077
m9l@ornl.gov

Loren J. Hauser
Computer Science & Mathematics Division
Oak Ridge National Laboratory
P.O. Box 2008, Bldg. 9211
Oak Ridge TN 37831-8077
hauserlj@bioax1.bio.ornl.gov

Edward C. Uberbacher
Computer Science & Mathematics Division
Oak Ridge National Laboratory
P.O. Box 2008, Bldg. 6010
Oak Ridge TN 37831-6364
ube@ornl.gov

Abstract

An important open problem in molecular biology is how to use computational methods to understand the structure and function of proteins given only their primary sequences. We describe and evaluate an original machine-learning approach to classifying protein sequences according to their structural folding class. Our work is novel in several respects: we use a set of protein classes that previously have not been used for classifying primary sequences, and we use a unique set of attributes to represent protein sequences to the learners. We evaluate our approach by measuring its ability to correctly classify proteins that were not in its training set. We compare our input representation to a commonly used input representation – amino acid composition – and show that our approach more accurately classifies proteins that have very limited homology to the sequences on which the systems are trained.

Introduction

A problem of fundamental importance in molecular biology is understanding the structure and function of the proteins found throughout nature. Currently, the growth of protein sequence databases is greatly outpacing the ability of biologists to characterize the proteins in these databases. Efficient computational methods for predicting protein structure and function are highly desirable because conventional laboratory methods, X-ray crystallography and NMR, are expensive and time-consuming. Currently, the best method for predicting the structure and function of a protein is to identify a homologous protein that has already been characterized. However, from current genome-sequencing efforts it appears that as many as half of the newly discovered proteins do not have corresponding, well-understood homologs (Fields *et al.* 1994). The goal of our research program, therefore, is to develop protein-classification methods that are not overly reliant on sequence homology, but instead represent the essential properties of analogous proteins that have similar folds. In this paper, we describe and evaluate a novel, machine-learning approach for classifying protein sequences according to their structural fold family. Our experiments indicate that our approach provides a promising alternative to homology-based methods for protein classification.

There is a wide variety of existing approaches for predicting structural or functional aspects of proteins given their primary (i.e., amino-acid) sequences. These approaches include tertiary structure prediction (e.g., Kolinski & Skolnick 1992), secondary structure prediction (e.g., Rost & Sander 1993), sequence homology searching (e.g., Pearson & Lipman 1988; Altschul *et al.* 1990), and classification according to folding class (Dubchak, Holbrook, & Kim 1993; Ferran, Ferrara, & Pflugfelder 1993; Metfessel *et al.* 1993; Wu *et al.* 1993; Nakashima & Nishikawa 1994; Reczko & Bohr 1994). Our approach falls into this latter category – protein classification – which itself encompasses a wide variety of methods. Existing protein classification methods vary on a number of dimensions including: the intended purposes of the systems; the level of abstraction of the protein classes; and whether or not the systems are able to discover their own classes. The approach we describe is novel in at least two respects: we use a set of protein classes that previously have not been used for classifying primary sequences, and we use a physical

*This research was supported by the Office of Health and Environmental Research, U.S. Department of Energy under Contract No. DE-AC05-84OR21400 with Martin Marietta Energy Systems, Inc.

set of attributes to represent protein sequences.

Our approach uses machine-learning methods to induce descriptions of sixteen protein-folding classes. The folding classes that we use were devised by Orengo et al. (1993) in a large-scale computational effort to cluster proteins according to their structural similarity. These classes comprise analogous, as well as homologous proteins. Whereas Orengo et al. used structural information to classify proteins, we are interested in classifying proteins when only their primary sequences are available. Thus, the role of the machine-learning algorithms in our approach is to induce mappings from primary sequences to folding classes. A key aspect of our method is the way in which we represent primary sequences to the learner. Unlike most protein-classification approaches, which represent proteins by their amino-acid composition, our method represents proteins using attributes that better capture the commonalities of analogous proteins. We empirically compare learning systems that use our input representation to learning systems that use amino-acid composition as their input representation. We show that our approach more accurately classifies proteins that have very limited homology to the sequences on which the systems are trained.

Table 1: **Protein class representation.** The middle column lists the classes we use in our classification method. The left column indicates class families, and the right column lists, for each class, the number of examples we use in our experiments.

family	class (fold group)	# examples
α	Globin	27
	Orthogonal	14
	EF Hand	5
	Up/Down	7
	Metal Rich	16
β	Orthogonal Barrel	5
	Greek Key	24
	Jelly Roll	5
	Complex Sandwich	7
	Trefoil	7
	Disulphide Rich	11
α/β	TIM Barrel	15
	Doubly Wound	26
$\alpha+\beta$	Mainly Alpha	9
	Sandwich	20
	Beta Open Sheet	14
total examples		212

Problem Representation

The task that we address is defined as follows: given the amino-acid sequence of a protein, assign the protein to one of a number of folding classes. This problem definition indicates that there are two fundamental issues in implementing a classification method for the task: determining the attributes that are to be used to represent protein sequences, and defining the classes that are to be predicted. The remainder of this section discusses how we address these two issues.

Class Representation

The classes that we use in our approach are the *fold groups* defined by Orengo et al. (1993) in their effort to identify protein-fold families. These classes represent proteins that have highly conserved structures, but often low sequence similarity. Thus, the classes represent analogous, as well as homologous, proteins. Table 1 lists the *fold groups* that we use as our classes, as well as the number of examples in each class that we use in our experiments, and whether each class falls into the α (primarily alpha), β (primarily beta), α/β (alternating α and β), or $\alpha+\beta$ (non-alternating α and β) family (Levitt & Chothia 1976).

The method that Orengo et al. used to define their *fold groups* involved four primary steps.

1. A set of proteins with known folds was assembled from the Brookhaven Protein Data Bank (Bernstein *et al.* 1977).
2. The proteins in this set were clustered according to sequence similarity. Using the Needleman-Wunsch algorithm (Needleman & Wunsch 1970), they performed pairwise comparisons on 1410 protein sequences selected in the previous step. They then used *single-linkage cluster analysis* to form clusters of related sequences. In single-linkage cluster analysis, two proteins, a and b, are assigned to the same cluster if there exists a chain of proteins linking a and b, such that each adjacent pair in the chain satisfies a defined measure of relatedness. Two proteins were deemed related, in this case, if their sequence identity was $\geq 35\%$. For small proteins, this threshold was adjusted using the equation of Sander and Schneider (Sander & Schneider 1991). Also, for proteins with 25–35% sequence identity, a significance test was used to determine if the proteins were to be considered related.
3. A representative protein was selected from each of the clusters formed in the previous step, and the resulting set of proteins was clustered according to structural similarity. Pairwise comparisons of proteins in this set were done using a variant of the Needleman-Wunsch algorithm that compared structural environments rather than primary sequences.
4. Multidimensional scaling was applied to the resulting structural homology matrix to form clusters of proteins with similar folds. The final *fold groups* were defined from this clustering by human interpretation, with the aid of schematic representations and topology diagrams.

In summary, Orengo et al. organized proteins with known structures into classes representing similar folds, but not necessarily similar primary sequences. Whereas, Orengo et al. developed their classes by clustering proteins according to structural similarity, we are interested in classifying protein sequences whose structures have not been determined. Obviously, their method is not applicable in such cases since it takes structural information as input. Our approach therefore uses machine-learning methods to induce mappings from primary sequences to folding classes.

Our data set is formed in the following manner: For each of the fold groups listed in Table 1, we select between one and five of the examples that are representatives (as listed by Orengo et al.) of the clusters formed in step 2. above. Note that this set contains sequences with only very limited homology. We then use each of these proteins as a query sequence to search the SWISS-PROT database (Bairoch & Boeckman 1992) for similar sequences. We use both BLAST (Altschul *et al.* 1990) and FASTA (Pearson & Lipman 1988) for sequence comparisons. As many as nine examples are extracted from each search and added to our data set to increase the number of examples for each fold.

Input Representation

In order to employ a machine-learning method in this task, it is necessary to define an input representation; that is, a scheme for representing the proteins that are given to the system. The input representation that we use for our protein classification approach involves a small number of attributes that can be readily computed from the primary sequence of a given protein. The attributes that we use are the following:

- **Average residue volume:** Using values that represent the volume of each amino acid's side group (Dickerson & Geis 1969), we calculate the average residue volume for a given sequence.
- **Charge composition:** We use three attributes to represent the fraction of residues in a given sequence that have positive charge, negative charge, and neutral charge (Lehninger, Nelson, & Cox 1993).
- **Polarity composition:** We use three attributes to represent the fraction of residues in a given sequence that are polar, apolar, and neutral (Lehninger, Nelson, & Cox 1993).
- **Predicted α-helix/β-sheet composition:** One of these attributes represents the fraction of the protein's residues that are predicted to occur in α-helices, the other represents the fraction that are predicted to occur in β-sheets. Note that both of these values are merely predictions, since the problem of calculating secondary structure from primary structure is exceptionally difficult itself. Our predictions are generated by a neural network that we trained using the data set of Qian and Sejnowski (1988). The trained network is scanned along the protein sequence, generating a prediction of α, β or *coil* for each residue. The number of α and β predictions are summed and then divided by the sequence length.
- **Isoelectric point:** Using the Wisconsin Sequence Analysis Package (version 6.0) (Devereux, Haeberli, & Smithies 1984), we calculate the isoelectric point of the given sequence.
- **Fourier transform of hydrophobicity function:** Using hydrophobicity values for each amino acid, we convert a given sequence into a one-dimensional hydrophobicity function, H. We calculate the modulus of the Fourier transform of this function as follows (Eisenberg, Weiss, & Terwilliger 1984):

$$\mu(\delta) = \left\{ \left[\sum_{n=1}^{N} H_n \sin(\delta n) \right]^2 + \left[\sum_{n=1}^{N} H_n \cos(\delta n) \right]^2 \right\}^{1/2}$$

  where $\mu(\delta)$ is the value for the periodicity with frequency δ, and n ranges over the residues in the sequence. We calculate this function at $1°$ intervals from $0°$ (corresponding to a period of infinity) to $180°$ (corresponding to a period of 2 residues). Finally, the six hydrophobicity attributes we use are computed by averaging values over each non-overlapping $30°$ interval in $[0°, 180°]$.

We normalize the values for the *volume*, *isoelectric*, and *hydrophobicity* attributes so that they fall in the range [0, 1]. Values for the other attributes naturally lie in this range.

Empirical Evaluation

The underlying hypotheses of our approach are:

- The folding class of a protein can be accurately predicted, given only its primary sequence.
- The best representation for this classification task is one that attempts to capture the commonalities of analogous proteins that are in the same folding class.

Many protein-classification studies have represented proteins by their amino-acid composition (Klein & Delisi 1986; Nakashima, Nishikawa, & Ooi 1986; Dubchak, Holbrook, & Kim 1993; Metfessel *et al.* 1993), or by some description of the amino-acid n-mers that occur in sequences (Ferran, Ferrara, & Pflugfelder 1993; Nakashima & Nishikawa 1994; Reczko & Bohr 1994). Our hypothesis is that this type of representation is not well suited to the classification of proteins that have no close relatives in existing databases, or that have no close relatives whose structure has been determined. We conjecture that methods trained using such a representation will perform poorly when asked to classify proteins that do not have homologs in the training set. Our view is that protein-classification methods should be aimed at characterizing proteins that do not have well-understood homologs.

In order to test our hypotheses, we present a number of experiments that evaluate our approach. First, we measure how well several machine-learning algorithms generalize[1] to unseen examples after learning to classify proteins using our problem representation. As a baseline for comparison, we also measure generalization for the same learning algorithms when amino-acid composition is used as the input representation. Our second experiment evaluates the relative contributions of the various attributes that comprise our input representation. Our third experiment tests the ability of systems trained using our representation to generalize to test cases for which there are no close relatives in the training set. This is a key experiment because our approach is motivated by the need to characterize proteins for which there are not any well-understood homologs. Finally, we demonstrate that the accuracy of our approach can be improved by having trained classifiers classify only examples for which they are confident in their predictions.

Measuring Generalization

The first task that we address in our experiments is to measure the accuracy of learners trained using our input and output representations. As a baseline for comparison, we also evaluate learners trained using amino-acid composition as their input representation. This representation has twenty attributes, each of which represents the fraction of a protein sequence that is composed of a particular amino acid.

We use several different learning algorithms to evaluate these two representations, since we do not know *a priori* which algorithm has the most appropriate inductive bias for each representation. We evaluate three inductive learning algorithms: C4.5 (Quinlan 1993), feed-forward neural networks (Rumelhart, Hinton, & Williams 1986), and k-nearest-neighbor classifiers (Cover & Hart 1967). We evaluate the suitability of these algorithms for the protein-classification task by estimating their generalization ability. In order to estimate generalization for each learning method, we conduct leave-one-out cross-validation runs.[2]

C4.5 is an algorithm for learning decision trees. The complexity of the trees induced by C4.5 can be controlled by pruning trees after learning. In our experiments, we run C4.5 both without pruning, and with pruning confidence levels ranging from 10% to 90%.

The neural networks that we use in our experiments are fully connected between layers, and have 3, 5, 10, 20 or no hidden units. We use the logistic activation function for hidden units, and the "softmax" activation function (Bridle 1989) for output units. The softmax function defines the activation of unit i as:

$$a_i = \frac{e^{\xi_i}}{\sum_n e^{\xi_n}}$$

where ξ_i is the net input to unit i, and n ranges over all of the output units. The networks are trained using the cross-entropy error function (Hinton 1989), and a conjugate-gradient learning method (Kramer & Sangiovanni-Vincentelli 1989), which obviates the need for learning-rate and momentum parameters. Networks are trained until either (1) they correctly classify all of the training-set examples, (2) they converge to a minimum, or (3) 1000 search directions have been tried. The networks have one output unit per class; the class associated with the most active unit is taken as the network's prediction for a given test example. Since the solution learned by a neural network is dependent upon its initial weight values, for all of our neural-network experiments we perform four cross-validation runs, using different initial weight settings each time.

For k-nearest-neighbor classifiers, we use a Euclidean distance metric to measure proximity. We construct classifiers that use values of k ranging from 1 to 10. The class predicted by a nearest-neighbor classifier is the plurality class of the k training examples that are nearest to a given test example. Ties are broken in favor of the nearest neighbor.

Table 2 reports leave-one-out accuracy for the best parameter settings for each learning method. The middle column lists the measured accuracy values for classifiers trained using our input representation. The right column list accuracy values for classifiers trained using amino-acid composition as their input representation. For both input representations, we found that pruning did not improve C4.5's generalization on this task, thus we report the accuracy of unpruned trees. For neural networks, the best results were obtained using 20 hidden units for our input representation, and no hidden units for the amino-acid representation. For the nearest-neighbor method, the best results were obtained by using k=3 for our input representation, and k=1 for the amino-acid representation.

Table 2: **Test-set accuracy using leave-one-out cross validation.** For each algorithm listed in the left column, the middle column lists the resulting test-set accuracy when our input representation is used. The right column lists test-set accuracy when amino-acid composition is used as the input representation.

learning method	**test-set accuracy**: our representation	amino-acid representation
C4.5	60.8%	49.1%
nearest-neighbor	80.7	76.9
neural networks	83.0	70.8

[1] *Generalization* refers to how accurately a system classifies examples that are not in its training set.

[2] In *leave-one-out cross-validation*, classifiers are trained on $n-1$ of the n available examples and then tested on the example left out. This process is repeated n times, so that each example is used as the testing example exactly once.

The accuracy values we list for the neural networks are averages over four cross-validation runs. The standard deviations for these averages are less than 0.1%, and thus we do not list them in the table. We omit standard deviations from the other tables in the paper for the same reason.

We draw two conclusions from this experiment. The first is that it is possible to classify protein primary sequences into Orengo et al.'s folding groups with high accuracy; the neural networks using our input representation were 83.0% accurate on test cases. The second conclusion we make is that our input representation is a better representation than amino-acid composition for this task. For all three learning methods, our representation resulted in superior generalization.

Evaluating The Input Representation

In our second experiment, we seek to understand how much the individual attributes that comprise our input representation contribute to the overall performance of the classifiers. To measure this, we conduct a series of leave-one-out cross-validation runs using nearest-neighbor classifiers (with k=3, the best value of k in the previous experiment), and input representations that contain only subsets of the attributes defined in the *Input Representation* section of this paper. First, we conduct leave-one-out runs in which the input representations consists only of individual attribute groups. For example, in one run we classify instances using only the *charge composition* attributes as our input representation. We also conduct leave-one-out runs in which we use all of the attributes except for a selected group. For example, in one run we classify examples using an input representation that consists of all of the attributes except for the *charge composition* attributes.

Table 3 reports the results of this experiment. The left column in the table lists the attribute groups. The middle column reports test-set accuracy for runs in which we use only a single attribute group. The right column reports test-set accuracy for cross-validation runs in which we use all of the attributes except for the indicated group. Note that the last row in the table lists the cross-validation accuracy that we measured for nearest-neighbor classifiers using our original input representation.

As the values in the middle column indicate, none of the attribute groups alone are as predictive as the entire attribute set. The values in the right column indicate that every attribute group contributes to the predictiveness of the original input representation. Although most of the accuracy values in this column are close to the accuracy of the entire attribute set, none of them equals or exceeds it. From these results we conclude that all of the attributes in our input representation make some contribution to the predictiveness of our classifiers.

Table 3: **Evaluating attribute predictiveness.** The table lists accuracy results for leave-one-out cross-validation runs with nearest-neighbor classifiers using selected attribute subsets. The middle column reports accuracy for input representations that use only the indicated attributes. The right column reports accuracy for input representations that omit the indicated attributes.

attributes	**using only**	**leaving out**
average residue volume	18.9%	78.3%
charge composition	21.7	78.8
polarity composition	31.6	79.2
α-helix/β-sheet composition	29.7	77.8
isoelectric point	15.1	76.9
FT of hydrophobicity	48.6	59.9
all attributes	80.7	—

Estimating the Role of Homology

Although the results presented in the first experiment indicate that we are able to classify proteins with high accuracy, they do not really address our primary concern; namely, that we want to accurately classify proteins when the classifier's training set does not contain sequences that are homologous to the test sequences. In this section, we present an experiment in which each test set has very limited homology to its corresponding training set.

As in previous experiments, we use cross-validation to estimate the accuracy of our classifiers. Unlike the previous experiments, however, the training and test sets in this experiment vary in size. The training and test sets for this experiment are formed by first partitioning the set of examples for each class into separate subsets, such that homologous proteins fall into a single subset. Recall that we formed our data set in the following manner: First, non-homologous sequences were selected from Orengo et al.'s data set. Sequences with 35% or greater sequence identity were considered homologous. For sequences with 25–35% sequence identity, a significance test was used to decide if the sequences were considered homologous. We then expanded our data set by using each of these sequences as a query sequence to find close relatives in the SWISS-PROT database. The resulting groups of homologous sequences correspond to the partitions we use in this experiment.

This partitioning of the examples allows us to do cross-validation runs in which we ensure that for every member of the test set there is *not* a close relative in the classifier's training set. A cross-validation run, in this experiment, involves using each of the subsets as the test set exactly once. The examples for four of the classes, α:*EF Hand*, α:*Up/down*, β:*Complex Sandwich*, and β:*Trefoil*, are not used in this experiment since all

Table 4: **Test-set accuracy factoring out the role of homology.** The training sets in this experiment do not contain sequences that are homologous to the test sequences.

learning method	test-set accuracy	
	our representation	amino-acid representation
C4.5	32.5%	25.5%
nearest-neighbor	36.6	30.6
neural networks	40.1	23.1

members of these classes share high sequence identity.

As in the first experiment, we run each algorithm several times using a range of parameter settings. C4.5 is run with pruning confidence levels ranging from 10% to 90%, and without pruning. We train neural networks with 3, 5, 10, 20 and no hidden units. Nearest-neighbor classifiers use values of k from 1 to 10.

Table 4 shows the results of this experiment for the best parameter settings for each learning method. As in our first experiment, we found that the best decision-tree generalization was achieved without pruning, As before, we also found that the best neural-network generalization was with networks that had 20 hidden units for our input representation, and no hidden units for the amino-acid representation. For the nearest-neighbor method, the best results were obtained by using k=4 for our input representation, and k=3 for the amino-acid representation.

The classification accuracy values reported for this experiment are all significantly worse than their counterparts in the first experiment. For the best classifier – a neural network trained using our input representation – generalization dropped from 83.0% to 40.1%. There are several reasons for this decrease in accuracy. One factor is that, due to our partitioning of the data, the training sets used in the second experiment are smaller than those used in the first. Some of the training sets had only one or two examples for some classes. The more important factor, however, is that homology plays a large role in the predictive ability of the classifiers described in our first experiment.

Although the results of this experiment are somewhat disappointing, they also lend support to our hypothesis that our input representation is better than one based on amino-acid composition. For all three learning algorithms, the classifiers trained using our input representation generalized significantly better than the classifiers using the amino-acid input representation. This result confirms that our representation embodies more of the commonalities of analogous proteins than does the amino-acid representation.

Table 5 shows per-class *sensitivity* values for our neural networks' predictions. The sensitivity of a set of predictions for class c is defined as the percentage of members of class c that are correctly identified as belonging to c. The table displays sensitivity values for both the first experiment (leave-one-out) and this one (no homologs). This table indicates that the accuracy of our predictions is not uniform across the classes, especially in the no-homologs experiment. As previously mentioned, poor performance for some classes is at least partly explained by sparse training sets. The poor predictability of some classes may also be due to the classes themselves being artificial constructs. It is important to keep in mind that the class structure itself was devised through a combination of automated clustering and human interpretation. Many of these classes encompass diverse groups of proteins, and in some cases, the class boundaries are rather arbitrary.

Table 5: **Neural-network sensitivity measurements by class.** The middle column shows per-class sensitivity values for the first experiment (leave-one-out cross validation). The right column shows sensitivity values for the third experiment (no test-set example has a homolog in the training set). Sensitivity is defined as the percentage of examples in a class that are correctly predicted.

class	leave-one-out sensitivity	no-homologs sensitivity
Globin	96.3%	96.3%
Orthogonal	71.4	7.1
EF Hand	80.0	—
Up/Down	85.7	—
Metal Rich	93.8	75.0
Orthog. Barrel	60.0	0.0
Greek Key	79.2	39.6
Jelly Roll	80.0	10.0
Complex Sand.	71.4	—
Trefoil	85.7	—
Disulphide Rich	72.7	0.0
TIM Barrel	80.0	43.3
Doubly Wound	92.3	53.8
Mainly Alpha	88.9	16.7
Sandwich	65.0	15.0
Beta Open Sheet	92.9	3.6

Improving Accuracy by Rejecting Examples

Although the classification accuracy values reported in the previous experiment are rather low, we have found that the accuracy of our classifiers can be improved by taking into account the confidence of their predictions. In this section, we describe an experiment in which we employ a strategy that is commonly used in the domain of handwritten character recognition: classifiers "reject" (i.e., do not classify) examples for which they cannot confidently predict a class (Le Cun *et al.* 1989).

In our fourth experiment, we evaluate neural networks and nearest neighbor classifiers using the same training and test sets as in the previous experiment (i.e., no test-set example has a homolog in the training set). We use single-nearest-neighbor classifiers and neural networks with 20 hidden units. For every test-set example, a confidence value is output along with the predicted class.

There are various algorithm-specific heuristics that can be used to estimate a classifier's confidence in a given prediction. For nearest-neighbor algorithms, we measure a prediction's confidence as how large we can make k while ensuring that the k nearest neighbors are unanimous in the class they predict. For neural networks, we measure confidence as the fraction:

$$\frac{a_i}{\sum_n a_n}$$

where a_i is the activation of the unit corresponding to the predicted class, and n ranges over all of the output units (actually the normalization is superfluous when the softmax activation function is used).

By establishing a threshold on the confidence values, we are able to have classifiers reject examples for which they are uncertain. Figure 1 displays generalization as a function of the percentage of examples rejected, for both neural networks and nearest-neighbor classifiers. The left edge of the graph represents the case where no examples are rejected. The right edge of the graph represents the case where 80% of the examples are rejected. For both methods, the curves climb steadily, meaning that the classifiers improve their accuracy as they throw out more examples.

The results of this experiment are interesting because they suggest that even if we cannot develop a classifier that is highly accurate for a wide range of proteins, we can perhaps develop a classifier that is highly accurate for certain classes of proteins, and is able to determine when test cases fall into these classes.

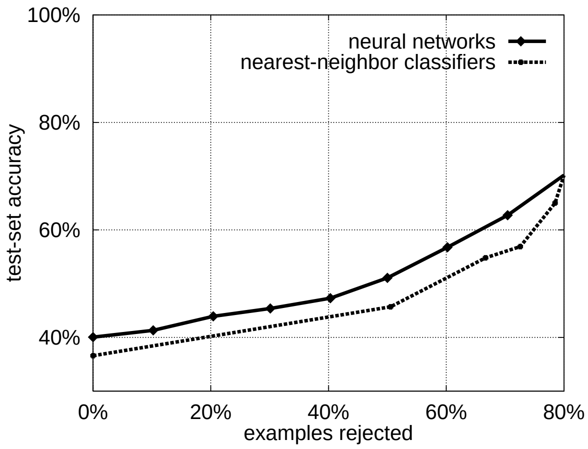

Figure 1: **Rejection curves.** The x-axis indicates the fraction of examples rejected. The y-axis indicates the corresponding accuracy.

Future Work and Conclusions

There are several open issues that we plan to explore in future research. These include: investigating alternative input representations, developing an alternative class structure, predicting folding classes for multi-domain proteins, and using distributed output representations. We discuss each of these in turn.

Although experiments presented in Section indicate that our input representation is superior to the commonly used amino-acid representation, we believe that our representation can be improved by incorporating additional attributes. For example, we plan to investigate attributes based on the Fourier transform of signals formed by residue volumes and charges.

Defining a set of protein-folding classes is a difficult problem in itself. To date we have used the folding groups identified by Orengo et al. as our classes. Some of these classes are rather arbitrarily defined, however, and hence difficult to predict. We plan to re-evaluate our current class structure to determine if some of the classes should be redefined, aggregated or discarded.

Another important and difficult issue to be addressed in our future research is how to predict the folding class of multi-domain proteins which do not fall completely into one of our existing classes. An accurate prediction for such a protein might involve labeling different domains of the protein with different classes. Since our approach enables subsequences of proteins to be represented and classified, the key problem to be solved in this task is how to parse protein sequences into subsequences. One possible approach is to generate secondary-structure predictions for a given protein, and then to use the predicted α-helix/β-sheet boundaries to suggest alternative parses.

Finally, we plan to investigate the utility of using a distributed output representation during learning. In a distributed representation, each of the problem classes is represented using a bit-string in which more than one bit is "on". A carefully engineered encoding scheme can result in significantly better generalization (Dietterich & Bakiri 1995).

We have presented a novel machine-learning approach to classifying proteins into folding groups. Our method uses attributes that can be easily computed from the primary sequence of a given protein. We have presented experiments that show that our approach is able to classify proteins with relatively high accuracy. We have also demonstrated that our input representation is superior to a representation based on amino-acid composition, especially when classifying proteins which have no homologs in the training set. The goal of our research program is to develop computational methods that are able to accurately classify proteins that have no well-understood homologs. We believe that the research presented herein represents a promising start towards this goal.

Acknowledgments. The authors would like to thank Carolyn Allex and the anonymous reviewers for their insightful comments on this paper.

References

Altschul, S. F.; Gish, W.; Miller, W.; Myers, E. W.; and Lipman, D. J. 1990. Basic local alignment search tool. *Journal of Molecular Biology* 215(3):403–410.

Bairoch, A., and Boeckman, B. 1992. The Swiss-Prot protein sequence data bank. *Nucleic Acids Research* 20:2019–2022.

Bernstein, F.; Koeyzle, T.; Williams, G.; Meyer, J.; Brice, M.; Rodgers, J.; Kennard, O.; Shimanouchi, T.; and Tatsumi, M. 1977. The protein data bank: A computer-based archival file for macromolecular structures. *Journal of Molecular Biology* 112:535–542.

Bridle, J. 1989. Probabilistic interpretation of feedforward classification network outputs, with relationships to pattern recognition. In Fogelman-Soulie, F., and Hérault, J., eds., *Neurocomputing: Algorithms, Architectures, and Applications*. New York, NY: Springer-Verlag.

Cover, T. M., and Hart, P. E. 1967. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory* 13(1):21–27.

Devereux, J.; Haeberli, P.; and Smithies, O. 1984. A comprehensive set of sequence analysis programs for the VAX. *Nucleic Acids Research* 12:387–395.

Dickerson, R. E., and Geis, I. 1969. *The structure and action of proteins*. Menlo Park, CA: W. A. Benjamin.

Dietterich, T. G., and Bakiri, G. 1995. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research* 2:263–286.

Dubchak, I.; Holbrook, S. R.; and Kim, S.-H. 1993. Prediction of protein folding class from amino acid composition. *Proteins: Structure, Function, and Genetics* 16:79–91.

Eisenberg, D.; Weiss, R. M.; and Terwilliger, T. C. 1984. The hydrophobic moment detects periodicity in protein hydrophobicity. *Proceedings of the National Academy of Sciences, USA* 81:140–144.

Ferran, E. A.; Ferrara, P.; and Pflugfelder, B. 1993. Protein classification using neural networks. In *Proceedings of the First International Conference on Intelligent Systems for Molecular Biology*, 127–135. Bethesda, MD: AAAI Press.

Fields, C.; Adams, M. D.; White, O.; and Venter, J. C. 1994. How many genes in the human genome? *Nature Genetics* 7(3):345–346.

Hinton, G. 1989. Connectionist learning procedures. *Artificial Intelligence* 40:185–234.

Klein, P., and Delisi, C. 1986. Prediction of protein structural class from the amino acid sequence. *Biopolymers* 25:1659–1672.

Kolinski, A., and Skolnick, J. 1992. Discretized model of proteins. I. Monte Carlo study of cooperativity in homopolypeptides. *Journal of Chemical Physics* 97(12):9412–9426.

Kramer, A. H., and Sangiovanni-Vincentelli, A. 1989. Efficient parallel learning algorithms for neural networks. In Touretzky, D., ed., *Advances in Neural Information Processing Systems*, volume 1. San Mateo, CA: Morgan Kaufmann. 40–48.

Le Cun, Y.; Boser, B.; Denker, J. S.; Henderson, D.; Howard, R. E.; Hubbard, W.; and Jackel, L. D. 1989. Handwritten digit recognition with a back-propagation network. In Touretzky, D., ed., *Advances in Neural Information Processing Systems*, volume 2. San Mateo, CA: Morgan Kaufmann.

Lehninger, A. L.; Nelson, D. L.; and Cox, M. M. 1993. *Principles of Biochemistry*. New York, NY: Worth Publishers.

Levitt, M., and Chothia, C. 1976. Structural patterns in globular proteins. *Nature (London)* 261:552–557.

Metfessel, B. A.; Saurugger, P. N.; Connelly, D. P.; and Rich, S. S. 1993. Cross-validation of protein structural class prediction using statistical clustering and neural networks. *Protein Science* 2:1171–1182.

Nakashima, H., and Nishikawa, K. 1994. Discrimination of intracellular and extracellular proteins using amino acid composition and residue-pair frequencies. *Journal of Molecular Biology* 238:54–61.

Nakashima, H.; Nishikawa, K.; and Ooi, T. 1986. The folding type of a protein is relevant to its amino acid composition. *Journal of Biochemistry (Tokyo)* 99:153–162.

Needleman, S. B., and Wunsch, C. D. 1970. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology* 48:443–453.

Orengo, C. A.; Flores, T. P.; Taylor, W. R.; and Thornton, J. M. 1993. Identification and classification of protein fold families. *Protein Engineering* 6(5):485–500.

Pearson, W. R., and Lipman, D. J. 1988. Improved tools for biological sequence comparison. *Proceedings of the National Academy of Sciences, USA* 85:2444–2448.

Qian, N., and Sejnowski, T. 1988. Predicting the secondary structure of globular proteins using neural network models. *Journal of Molecular Biology* 202:865–884.

Quinlan, J. R. 1993. *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann.

Reczko, M., and Bohr, H. 1994. The DEF data base of sequence based protein fold class predictions. *Nucleic Acids Research* 22(17):3616–3619.

Rost, B., and Sander, C. 1993. Prediction of protein secondary structure at better than 70% accuracy. *Journal of Molecular Biology* 232:584–599.

Rumelhart, D.; Hinton, G.; and Williams, R. 1986. Learning internal representations by error propagation. In Rumelhart, D., and McClelland, J., eds., *Parallel Distributed Processing: Explorations in the microstructure of cognition. Volume 1: Foundations.* Cambridge, MA: MIT Press. 318–363.

Sander, C., and Schneider, R. 1991. Database of homology-derived protein structures and the structural meaning of sequence alignment. *Proteins: Structure, Function and Genetics* 9:56–68.

Wu, C.; Berry, M.; Fung, Y.-S.; and McLarty, J. 1993. Neural networks for molecular sequence classification. In *Proceedings of the First International Conference on Intelligent Systems for Molecular Biology*, 429–437. Bethesda, MD: AAAI Press.