Machine Learning, 21, 103–124 (1995)
© 1995 Kluwer Academic Publishers, Boston. Manufactured in The Netherlands.

Use of Adaptive Networks to Define Highly Predictable Protein Secondary–Structure Classes

ALAN S. LAPEDES asl@t13.lanl.gov
Complex Systems Group (T13)
LANL, Los Alamos, NM 87545
and
The Santa Fe Institute, Santa Fe, NM

EVAN W. STEEG steeg@cs.toronto.edu
Department of Computer Science
University of Toronto
Toronto, ON M5S 1A4
Canada

ROBERT M. FARBER rmf@t13.lanl.gov
Complex Systems Group (T13)
LANL, Los Alamos, NM 87545
and
The Santa Fe Institute, Santa Fe, NM

Editors: Jude Shavlik, Lawrence Hunter, and David Searls

Abstract. We present an adaptive, neural network method that determines *new* classes of protein secondary structure that are significantly more predictable from local amino–acid sequence than conventional classifications. Accurate prediction of the conventional secondary–structure classes, alpha-helix, beta-strand, and coil, from primary sequence has long been an important problem in computational molecular biology, with many ramifications, including multiple–sequence alignment, prediction of functionally important regions of proteins, and prediction of tertiary structure from primary sequence. The algorithm presented here uses adaptive networks to simultaneously examine both sequence and structure data, as available from, for example, the Brookhaven Protein Database, and to determine new secondary–structure classes that can be predicted from sequence with high accuracy. These new classes have both similarities to, and differences from, conventional secondary-structure classes. They represent a new, nontrivial classification of protein secondary structure that is predictable from primary sequence.

Keywords: Predictable classes, adaptive systems, correlation, mutual information, coupled networks, protein secondary structure, clustering

1. Introduction

The conventional classes of protein secondary structure, alpha-helix and beta-strand, were first introduced in 1951 by Linus Pauling and Robert Corey (Pauling *et al.*, 1951) on the basis of molecular modeling. Two periodic peptide structures, which they called alpha-helix and beta-strand, could be built as detailed molecular models that satisfied experimentally–determined constraints on bond angles and distances. The alpha-helix secondary structure was later observed in the x-ray diffraction reconstruction of the hemoglobin protein (Perutz, 1951). Examples of beta-strand secondary structure may be found in, for example, silk. These two classes of structure are visually quite apparent in modern molecular graphics representations of x-ray diffraction models of proteins. It has long been an outstanding problem of computational molecular biology to be able to predict these classes of secondary structure from the amino–acid sequence. Before attempting to predict such structures, however, it is necessary to have an unambiguous, algorithmic definition of the structures, as opposed to mere visual identification.

As more crystallographically–determined protein structure models became available, their secondary structures were annotated (often, only on the basis of visual inspection) as apparent helices and strands. To remove confusion and conflicting classifications, formal definitions of alpha-helices, beta-strands, and other secondary–structure classes were constructed on the basis of certain characteristics of the local geometry of proteins. For example, a hierarchical definition that is in widespread use today first defines potential hydrogen bonds based on structural coordinates, and subsequently defines alpha-helices and beta-strands as particular patterns of the potential hydrogen bonds (Kabsch & Sander, 1983).

In this paper we will consider, as a valid definition and generalization of protein "secondary structure", *any* classification of protein structure that can be defined using only local "windows" of structural information about the protein. Such structural information could be, for example, the classic $\Phi\Psi$ angles (Schulz & Schirmer, 1979) that describe the relative orientation of peptide units along the protein backbone, or any other representation of local backbone structure. (See Figure 1.) We define a classification of local structure into "secondary–structure classes" to be the result of any algorithm that uses a representation of local structure as input, and which produces discrete classification labels as output. This is a very general definition of local secondary structure that subsumes all previous definitions. The goal of this paper is to use neural networks to define such generalized secondary–structure classes, with the important restriction that the new secondary–structure classes be highly predictable from amino-acid sequence.

It is the extra restriction of predictability that we impose which distinguishes this work from other work (Hunter & States, 1992; Unger *et al.*, 1989) that deals only with representations of the local structure, and that does not simultaneously consider the predictability of the structure from sequence. It is clearly possible to consider only local shape information and to employ a clustering algorithm to define new secondary–structure classes based on these local shapes. However, we employ an algorithm that simultaneously classifies shapes and selects those classes that are highly predictable from sequence.

The classifications we develop are more predictable than the standard classifications (Pauling *et al.*, 1951; Kabsch & Sander, 1983) which were used in previous machine learning projects, as well as in other analyses of protein shape. These new, predictable classes of secondary structure do bear some relation to the conventional category of "helix" but also display significant differences. More analysis will be required to fully understand the new classes. It is also presently an open question whether these new classes of secondary structure will be more useful in protein structure analysis than conventionally defined classes, or than classes defined by structural clustering alone. It

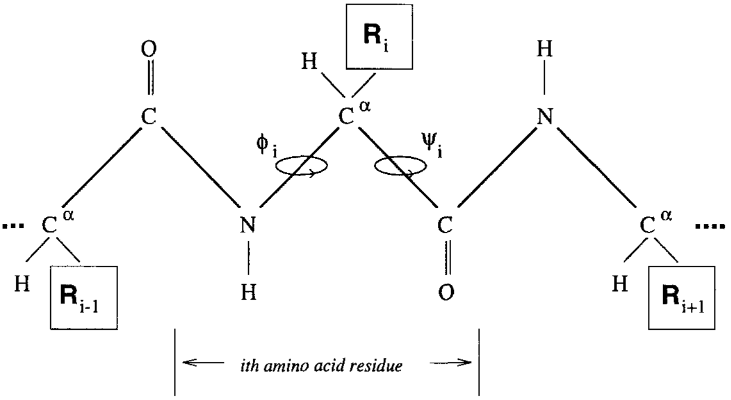

Figure 1. This schematic of a polypeptide (protein) backbone structure illustrates the significance of the ϕ and ψ angles. Each amino acid molecule, or residue, in the chain features an *R-group*, or *side–chain*, characteristic of the particular amino acid occurring in that position. The R-group is shown bonded with the *α-carbon* atom. The covalent bonds comprising the backbone, and the bond angles, are fairly rigid. However, considerable rotation is possible *around* certain bonds. The angles ϕ_i and ψ_i measure the torsion about the rotationally permissive bonds in the backbone of the ith residue. The backbone structure of a protein may be specified by the ordered list of coordinates in 3-space of the α-carbons of the residues, or by a two-dimensional table of Euclidean distances between α-carbons, for all pairs of residues, or by a list of the ϕ, ψ angles for all residues.

is promising, however, that the general methodology we advocate has produced new classes of secondary structure that have some relation to conventional classes, and yet are more predictable from amino-acid sequence.

2. Prediction of Conventional Secondary–Structure Classes

A major reason that prediction of secondary structure is of interest is that a successful prediction of secondary structure from amino-acid sequence may be used in tertiary–structure prediction algorithms to constrain their search space (Skolnick & Kolinski, 1991). Tertiary structure refers to the detailed three-dimensional conformation that proteins adopt when they fold into their natural shape. The ability to calculate tertiary structure from amino-acid sequence has many important ramifications, including the ability to predict functional properties of a protein given only its amino-acid sequence. One approach to prediction of the full, detailed tertiary structure of a protein begins by first predicting the local secondary–structure class for each amino-acid. It is possible to show that accurate secondary–structure information is extremely helpful in tertiary–structure prediction. For example, Skolnick (1991) has found that biasing amino acids towards assuming the correct, measured secondary structure, when coupled to his global tertiary–structure prediction algorithms, greatly increases the agreement of the global tertiary–structure prediction with the experimentally–determined structure. However, his test of the value of knowing the secondary–structure classes used the *actual* secondary–structure classes as determined from experimental data, and not error-prone algorithmic *predictions* of secondary–structure classes from amino-acid sequence. His method, and others, are not generally successful if they attempt to use predictions of secondary–structure classes at current levels of accuracy.

A widely adopted definition of protein secondary–structure classes is due to Kabsch and Sander (1983). It has become conventional to use the Kabsch and Sander definition to define, via local structural information, three classes of secondary structure: alpha-helix, beta-strand, and a default class called random coil. The Kabsch and Sander alpha-helix and beta-strand classification captures in large part the classification first introduced by Pauling and Corey. · Structures of many proteins have now been determined using x-ray diffraction and many examples of alpha-helices, beta-strands, and random coils are contained in databases such as the Brookhaven Protein Database (Abola *et al.* 1987).

There have been numerous attempts to predict locally–defined secondary–structure classes using only a local window of sequence information. The prediction methodology ranges from a combination of statistical and rule-based methods (Chou & Fasman, 1978) to neural network methods (Qian & Sejnowski, 1988; Maclin & Shavlik, 1992; Kneller *et al.*, 1990; Stolorz *et al.*, 1992). Figure 2a illustrates the Kabsch and Sander program defining secondary–structure classes, depicted as a "black box" on the right, and also a neural network that attempts to learn the secondary–structure classes from the amino–acid sequences, on the left. The Kabsch and Sander "black box" first defines hydrogen–bonding patterns from the structural information, and then uses the hydrogen–bonding patterns to define classes of secondary structure. This picture represents the standard approach to training a neural network to classify secondary structure from amino-acid sequence (Stolorz *et al.*, 1992). A local window of structure information obtained from, for example, x-ray diffraction data in the Brookhaven database, is input to the right–hand Kabsch and Sander black box. The box outputs the secondary–structure class of the fragment, using the Kabsch and Sander definitions. For example, if one were dichotomizing all the windows of structure information into "alpha-helix" and "not-alpha-helix", then the right–hand box will emit a "1" if the fragment is alpha-helix, and emit a "0" otherwise. The left–hand neural network "sees" the corresponding window of sequence information as input, and attempts to adjust its synaptic weights so that the output neuron of the neural network agrees with the output state of the Kabsch and Sander black box. Hence, if the input sequence adopts an alpha-helix state according to Kabsch and Sander, then the output neuron of the network should change state to "1". Conversely, an input sequence fragment not in an alpha-helix should cause the state of the output neuron to change to "0".

We consider in this exposition the definition, and prediction from sequence, of just two classes of structure. The extension to multiple classes is not difficult, but will not be made explicit here for reasons of clarity. We will not discuss details concerning construction of a representative training set, or details of conventional neural network training algorithms, such as backpropagation. These are well-studied subjects that are addressed in, for example, (Stolorz *et al.*, 1992) in the context of protein secondary–structure prediction. We note in passing that one can employ complicated network architectures containing many output neurons (for example, three output neurons for predicting alpha-helix, beta-strand, random coil), or many hidden units, etc. (c.f. Stolorz *et al.*, 1992; Qian & Sejnowski, 1988; Kneller *et al.*, 1990). However, explanatory figures presented in the next section employ only one output unit per net, and no hidden units, for clarity.

3. Definition and Prediction of New Secondary–Structure Classes

The key ideas of this section are contained in Figures 2b and 2c. In Figure 2b the right–hand black box implementing the Kabsch and Sander rules is replaced by a second neural network. This right–hand neural network therefore sees a window of structural information, while the left–hand neural network sees the corresponding window of sequence information. Note that the right–hand neural network can implement extremely general definitions of secondary structure. For example, if the weights in the right–hand network are set to arbitrary values, then the right–hand network will correspondingly produce an arbitrary classification of the structures that are input to it. On the other hand, one could train the weights of the right–hand network to perform structure classification according to, say, the Kabsch and Sander rules. To demonstrate the generality of the procedure we have done the latter, and have successfully captured the Kabsch and Sander structural definitions in the right–hand network with high accuracy. This explicitly demonstrates that neural networks are capable of representing rules of the complexity of the original Kabsch and Sander rules.

The representation of the structure data in the right–hand network uses $\Phi\Psi$ angles. The right–hand network sees a window of $\Phi\Psi$ angles corresponding to the window of amino acids in the left–hand network. Problems due to the angular periodicity of the $\Phi\Psi$ angles (that is, 360 degrees and 0 degrees are different numbers, but represent the same angle) are eliminated by utilizing both the sin and cos of each angle. Thus, for input to the network, each pair of $\Phi\Psi$ angles are replaced by $cos(\Phi), sin(\Phi), cos(\Psi), sin(\Psi)$. The representation of the amino acids in the left–hand network is the usual unary representation employing twenty bits per amino acid. We follow the conventional interpretation of the window approach to prediction: the prediction of structure is made for the *central* amino–acid residue in the window, with residues in the leading and trailing half-windows providing contextual information. Experiments described in this paper did not use, as some groups have used, a special twenty–first bit to represent positions in a window extending past the ends of a protein.

The following three points summarize the major themes of this paper:

Point (1): *One can replace the right–hand black box of Figure 2a with a neural network (see Figure 2b). A neural network on the right–hand side is an equally valid implementation of a set of rules defining secondary structure as is a traditional piece of software.* We have explicitly demonstrated this by training a neural network to reproduce the Kabsch and Sander rules with high accuracy.

Point (2): *The right–hand network need not be restricted to implementing the Kabsch and Sander rules for secondary structure.* The right–hand neural network is capable of representing a very general set of rules, of which the Kabsch and Sander rules are but one choice.

To define new rules one merely changes the synaptic weights. Arbitrary synaptic weights would define arbitrary rules, and there would be little chance that these new classes would be either predictable or meaningful.

Point (3): *A requirement on the rules is needed. A useful requirement is that the "secondary–structure" classes defined by the right–hand network be predictable from the corresponding amino-acid sequence of the left–hand network.*

In other words, we require that the synaptic weights be chosen so that the output of the left–hand network and the output of the right–hand network agree for each sequence–structure pair that is input to the two networks. This is illustrated in Figure 2b.

To achieve this, both networks are trained *simultaneously*, starting from random initial weights in each net, under the sole constraint that the outputs of the two networks agree for each pattern — or for as many patterns as is possible — in the training set. The mathematical implementation of this constraint is described in various versions below, and is illustrated in Figure 2c. The two networks become mutually self-supervising and implement a type of unsupervised clustering on the data. This co-evolution of the two networks is clearly a more difficult computational problem than the conventional approach (Figure 2a) that employs fixed targets. Each network now chases a moving target during training, and numerical difficulties in the form of local minima occur. These difficulties are surmountable, and new definitions of secondary structure may be found. This procedure is therefore a general, effective method of evolving *predictable* secondary–structure classifications of experimental data. The goal of this research is to use two mutually self-supervised networks to define new classes of protein secondary structure that are more predictable from sequence than the standard classes of alpha-helix, beta-strand and coil.

4. Constraining the Two Networks to Agree

A naive method to require that the two networks agree is obtained by analogy to the conventional neural network training algorithm, backpropagation with least mean squared–error (LMS) (Rumelhart & McClelland, 1986). In that algorithm, one performs gradient descent in the synaptic weights of a network in order to minimize the error function, E:

$$E = \sum_p (t^{(p)} - LeftO^{(p)})^2 \tag{1}$$

where $t^{(p)}$ is a specified target output value for the p^{th} pattern, and $LeftO^{(p)}$ is the output of the sequence neural network for the p^{th} pattern (see Figure 2a). $LeftO^{(p)}$ is a function of the synaptic weights. Gradient descent in the synaptic weights will decrease the error, E, evaluated on the training set by forcing the output of the network, $LeftO^{(p)}$ to agree with the target output, $t^{(p)}$, for each pattern.

Note that in Figure 2a the target values for the left–hand network are given by the fixed rules implemented in the right–hand Kabsch and Sander black box. These target values of conventional secondary structure, $t^{(p)}$, are fixed constants, i.e., either "0" or "1" as defined by the Kabsch and Sander definitions for each pattern of structural information.

As shown in Figure 2b, one might consider using the same error function, Eqn. (1), but replacing the previously fixed target values for each pattern by the variable output of the right–hand network. Hence the new error function, whose minimization would enforce agreement of the left–hand and right–hand networks is

$$E = \sum_{p} (LeftO^{(p)} - RightO^{(p)})^2 \tag{2}$$

The difficulty with this is that there is a trivial way for the two nets to agree. They merely need to decrease their synaptic weights from the inputs to zero, so each will stay in the "0" state, regardless of the input pattern (it is also possible to have each stay in the "1" state). The outputs of the two nets would then remain either "on" or "off" regardless of the input data and would trivially agree, thereby minimizing E of Eqn. (2). One might consider adding a variance term to Eqn. (2) to require the networks to respond to their inputs (i.e., to impose variation in the network outputs as the input patterns change). While this work was in progress we received a preprint by Schmidhuber (Schmidhuber, 1992) who essentially implemented Eqn. (2) with variance, in a totally different context. In our hands this measure was quite susceptible to local minima, whereas the measures we derive below to enforce agreement were less susceptible.

One way to impose the required agreement between the outputs of the two networks is to require that they co-vary when viewed as a stream of real numbers. (See Figure 2c.) Therefore, one can maximize the correlation, ρ, between the left–hand and right–hand network outputs. The standard correlation measure between two objects, $LeftO^{(p)}$ and $RightO^{(p)}$ is:

$$\rho = \sum_{p} (LeftO^{(p)} - \overline{LeftO})(RightO^{(p)} - \overline{RightO}) \tag{3}$$

where $\overline{LeftO}$ denotes the mean of the left net's outputs over the training set, and respectively for the right net. The expression ρ is zero if there is no variation, and is maximized if there is simultaneously both individual variation and joint agreement. In our situation it is equally desirable to have the networks maximally anti-correlated as it is for them to be correlated. (Whether the networks choose correlation, or anti-correlation, is evident from the behavior on the training set.) Hence the minimization of $E = -\rho^2$ would ensure that the outputs are maximally correlated (or anti-correlated).

Alternatively, since one ultimately measures predictive performance on the basis of the Mathews correlation coefficient (see, for example, Stolorz *et al.*, 1992), it is also reasonable to simultaneously train the two networks to maximize this measure. The Mathews coefficient, C_i, for the i^{th} state or class, is defined as:

$$C_i = \frac{p_i n_i - u_i o_i}{[(n_i + u_i)(n_i + o_i)(p_i + u_i)(p_i + o_i)]^{1/2}} \tag{4}$$

where p_i is the number of examples where the left–hand network and right–hand network both predict class i, n_i is the number of examples where neither network predicts i, u_i counts the examples where the left network predicts i and the right network does not, and o_i counts the reverse. Minimizing $E = -{C_i}^2$ maximizes C_i.

Other training measures forcing agreement of the left and right networks may be used. Particularly suitable for the situation of many outputs (i.e., more than two–class discrimination) is "mutual information". Use of mutual information in this context is related to the IMAX algorithm for unsupervised detection of regularities across spatial or temporal data (Becker & Hinton, 1992). The mutual information is defined as

$$M = \sum_{i,j} p_{ij} \log \frac{p_{ij}}{p_{i.}p_{.j}} \tag{5}$$

where p_{ij} is the joint probability of occurrence of the states of the left and right networks. The quantity $p_{i.}$ is defined as $p_{i.} = \sum_j p_{ij}$ and the quantity $p_{.j}$ is defined as $p_{.j} = \sum_i p_{ij}$. (In previous work (Stolorz *et al.*, 1992) we showed how p_{ij} , $p_{i.}$, and $p_{.j}$ may be defined in terms of neural networks.) Minimizing $E = -M$ maximizes M. While M has many desirable properties as a measure of agreement between two or more variables (Stolorz *et al.*, 1992; Farber *et al.*, 1992; Lapedes *et al.*, 1990; Korber *et al.*, 1993), our preliminary simulations show that maximizing M is often prone to poor local maxima.

Finally, an alternative to using mutual information for multi-class, as opposed to dichotomous classification, is the Pearson correlation coefficient, X^2. This is defined in terms of p_{ij} as

$$X^2 = \sum_{i,j} \frac{(p_{ij} - p_{i.}p_{.j})^2}{p_{i.}p_{.j}} \tag{6}$$

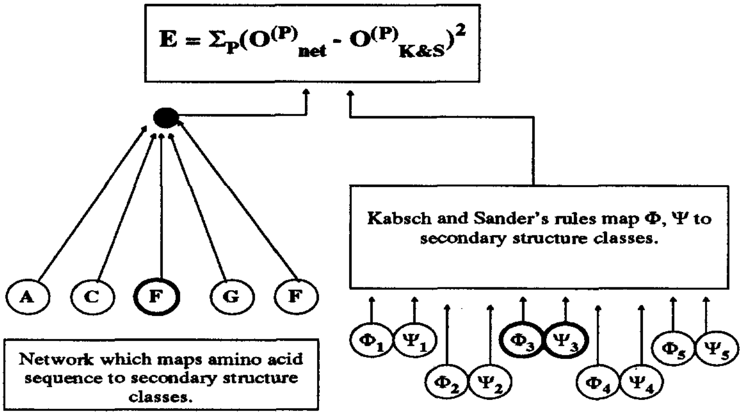
$E = \Sigma_P(O^{(P)}_{net} - O^{(P)}_{K\&S})^2$
Kabsch and Sander's rules map Φ, Ψ to secondary structure classes.
A
C
F
G
F
Network which maps amino acid sequence to secondary structure classes.
Φ_1
Ψ_1
Φ_2
Ψ_2
Φ_3
Ψ_3
Φ_4
Ψ_4
Φ_5
Ψ_5

Figure 2a. Conventional neural network training for prediction of conventional secondary–structure classes.

We note that it can be shown that both the correlation-based objective function and a variant of the mutual information function are nonlinear extensions of a standard statistical

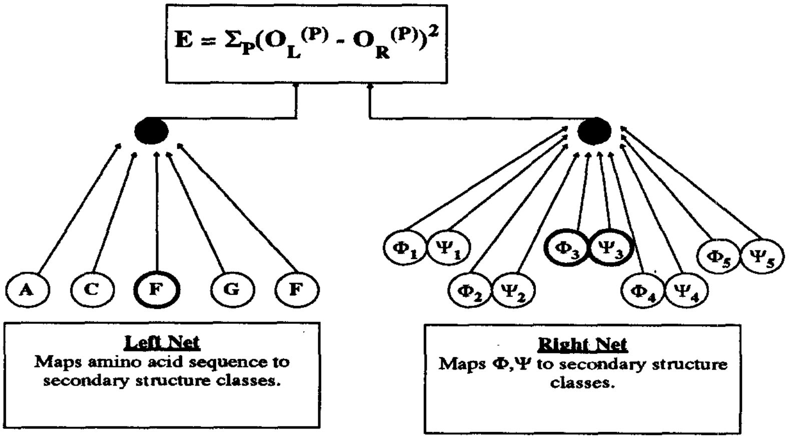
E = Σ_P(O_L^(P) - O_R^(P))^2
A
C
F
G
F
Φ1
Ψ1
Φ2
Ψ2
Φ3
Ψ3
Φ4
Ψ4
Φ5
Ψ5
Left Net
Maps amino acid sequence to secondary structure classes.
Right Net
Maps Φ,Ψ to secondary structure classes.

Figure 2b. Naive attempt at training coupled neural networks to evolve new, predictable secondary–structure classes. The left–hand network learns a mapping from local primary structure (amino-acid sequence) to secondary structure, while the right–hand network learns to map tertiary structure (as represented by ϕ, ψ angles) to secondary structure. Their joint task is to learn a mutually–predictable definition of secondary structure. However, the use of the standard LMS error measure can lead to the definition of trivial classes.

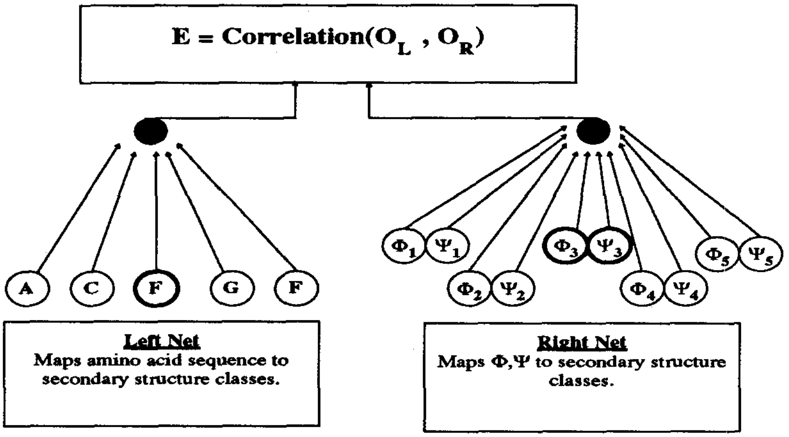
E = Correlation(O_L , O_R)
A
C
F
G
F
Φ1
Ψ1
Φ2
Ψ2
Φ3
Ψ3
Φ4
Ψ4
Φ5
Ψ5
Left Net
Maps amino acid sequence to secondary structure classes.
Right Net
Maps Φ,Ψ to secondary structure classes.

Figure 2c. Use of a correlation–based agreement measure to train coupled neural networks. Such training can produce novel and highly–predictable protein secondary–structure classifications.

method, Canonical Correlation Analysis (CCA) (Mardia *et al.*, 1979). CCA is used to find linear functions L_1, L_2 of two multi-dimensional variables, x_a and x_b, that maximize the correlation $\rho(L_1(x_a), L_2(x_b))$. Our neural networks and objective functions extend CCA by replacing simple linear combinations with nonlinear transformations (Becker, 1992).

The objective functions described above are defined either in terms of the networks' outputs or in terms of frequencies of the possible classification decisions observed over the example set. It is necessary to relate the two, in order to understand the learning task and the performance (prediction) task. There are many known ways to translate output unit activations O_i into classifications, including, for the simple 2–class case under discussion:

1. "soft classes", with $Prob(example_p \in CLASS1) = O^{(p)}$, where $O \in [0,1]$;
2. "hard classes", with $example_p \in CLASS1$ iff $O^{(p)} > 0.5$;
3. "hard classes', with $example_p \in CLASS1$ iff $O^{(p)} > \bar{O}$, where $\bar{O}$ is the mean output over all examples.

4.1. Practical Challenges in Optimizing the Agreement

It is useful to separate clearly the *objectives* of a learning task, and hence the objective functions, from the numerical methods used — and problems encountered, and compromises made — in pursuing these objectives. In our work, the objective functions which have the best theoretical justification pose implementational challenges.

Intuitively, it is not surprising that local minima are a significant problem for an investigation of this type that involves two co-varying networks. In contrast to the usual situation in backpropagation, in which a single network is trained so that each train-set pattern is matched to its fixed, target output value, here we are training the output of one network to match the output of a second network. Hence, one network is providing a target, in fact a moving target, for the other network. It is not surprising that numerical problems in the guise of numerous local minima should occur. Various solutions to the local minima problem, such as adding a small amount of noise during training, may be possible; however, we found that moving to a gradient–less procedure (in Section 6.1, Experiment 1) or to a smoother objective function (in Section 6.2, Experiment 2) was sufficiently effective to complete these investigations. The price paid in using a gradient–less procedure is speed. The simulations reported in Experiment 1 were performed on a CM5 Connection Machine, and typically required a few hours for training a single network to an acceptable train-set accuracy.

The mutual information function, in particular, often gets trapped quickly in local minima when evolved from random initial conditions. More success was obtained with the other objective functions. Of course, it is possible to "gang" together objective functions: one could start training from random initial conditions using, for example, the correlation objective function (which seems less susceptible to local minima than the mutual information function), and then finish with the mutual information function. We have not exhaustively investigated these strategies, and usually just chose new initial conditions if an uninteresting, shallow local minimum was encountered.

Our simulations indicate that X^2, C_i and ρ are all less susceptible to local minima than M. However, these other objective functions suffer the defect that predictability is emphasized at the expense of utility. In other words, they can be maximal for the peculiar situation where a structural class is defined that occurs very rarely in the data, but when it occurs, it is predicted perfectly by the other network. The utility of this classification is therefore degraded by the fact that the predictable class only occurs rarely. Fortunately, this effect did not cause serious difficulties in the simulations we performed. Our best results to date have been obtained using the Mathews objective function with "hard" classification decisions and a gradient–less optimization method.

Further details on these optimization and class-implementation issues are presented for each set of experiments in the Results section.

5. Clustering: Secondary Structure as a Representation

The different methods described above, defined by use of different objective functions, are related by a common, underlying type of *clustering* that they impose on the data. Assume one is given a training set, considered as a set of points $x_s = (a_s, z_s)$, where $a_s \in A$, $z_s \in Z$, where A is a space of short subsequences of amino acids, and where Z is a space of *local* tertiary–structure parameters ($\Phi\Psi$ angles) corresponding to the amino–acid residues. The left–hand and right–hand networks, in any of the two-network schemes described above, may be viewed as nonlinearly transforming the spaces A and Z, mapping points into new spaces A', Z', respectively, such that clusters are formed. These structural clusters are interpreted as secondary–structure classes (see Figure 3).

The objective functions that force "agreement" between outputs of the left and right nets enforce a compatibility constraint between the clusterings in the two different spaces, as well as enforcing other constraints on the sizes and shapes of the clusters. In general, the compatibility goal is achieved to the extent that neighboring points in sequence space A' are also neighbors in the structure space Z', and vice versa. That is, for $x_r = (a_r, z_r)$ and $x_s = (a_s, z_s)$, if a_r and a_s are in the same cluster, then z_r and z_s should be in the same cluster, and conversely (as shown in Figure 3). To the extent that this constraint holds, we say that the structure classification is compatible with the sequence classification, or in other words that the structure classes are predictable from sequence.

Other recent work in defining new structural classes includes efforts by groups taking an AI point of view. Here, protein secondary structure is considered to be a convenient and important representational scheme for encoding knowledge about an intermediate level of abstraction within a multi-level protein–analysis system. This work has focused on finding "intrinsic" clusters in local stretches of tertiary structure (Hunter & States, 1992), that is, classes which reflect objectively definable features as opposed to subjective visual or historical judgements. Although these projects have produced secondary–structure classes that meet several criteria for utility in further levels of protein analysis, they have not explicitly attempted to find classes that are highly predictable, which is clearly an important criterion for use in protein structure/function prediction systems.

In addition to efforts to find intrinsic clusters in the space of structural features, there has also been work aimed at finding intrinsic clusters in sequence space (see, for example, (Delorme & Henaut, 1988; Fitch, 1981; Hunter *et al.*, 1992; Zhang & Waltz, 1993).) The primary difference between either the structure–based clustering work, or the sequence–based clustering work, and our work, is that we impose the condition that

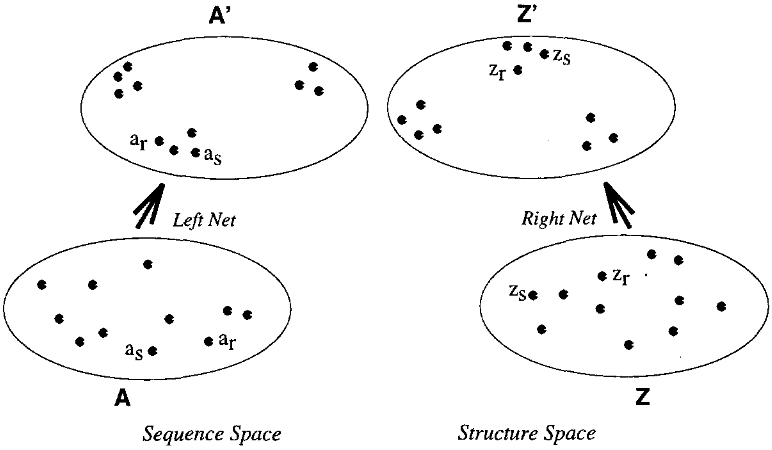

Figure 3. Compatible clusterings in sequence and structure spaces.

the sequence/structure clusters implicitly defined by the two networks are compatible, in the sense that at least one of the measures discussed in the previous section is optimized. In other words, we do not separately seek clustering in the structure space, and in the sequence space, and leave to later the investigation of whether these clusterings are correlated. Instead, we impose from the beginning that the sequence clusters must be correlated with the structure clusters, i.e., that one is predictable from the other.

It may be useful to locate each such approach to sequence-based or structure-based clustering in a hierarchy of possible approaches to combining sequence and structural information:

1. At the bottom is the basic clustering of local tertiary–structural (or amino-acid sequence) classes, according to some Bayesian or similar criteria, and ignoring sequence (or structure) space (Hunter & States, 1992; Zhang & Waltz, 1993).

2. The next level is to *assume* either the sequence or structure classification, and make these class designations a *feature* in the representation of the input examples in the other space; and then to cluster within that other space. In other words, one is still clustering within only one space, but this clustering is constrained, to some extent, to respect the (fixed) class structure in the other space. For example, the standard backpropagation training for the traditional secondary–structure prediction task may be viewed as a clustering of short amino-acid subsequences under the constraint that the clusters agree with the helix, strand, and coil structure classes.

3. The work presented in this paper is at the next level in the hierarchy. This level corresponds to clustering within the two spaces simultaneously, with the evolving class structure in each space constraining the evolution of the other.

6. Experiments and Results

Reported here are the results from two sets of experiments with the coupled neural networks system described above. In Experiment 1 a number of training runs were performed for a very computation–intensive version of our method, from which our best results were obtained. In Experiment 2 a much larger number of training runs were performed with a very fast variant of the algorithm, permitting a more detailed study of the dynamics and results of the mutually-supervised classification methodology. The best results to date, for each of the two sets of experiments, are presented below.

The criterion for choosing the "best" result in Experiment 1 was based on how well the networks could be trained using different objective functions and not by selecting the algorithm and architecture that performed best on the predict set. If one chooses the architecture and training termination time based on the predict set, then contamination of the train and predict sets occurs, and prediction accuracy can be over-estimated. Use of a cross-validation set to choose the network architecture, as well as the optimal amount of training, is an acceptable procedure. However, since the training times were already quite long (hours of CM5 time) in Experiment 1, it was not possible to perform a full cross-validation study. The difference between accuracies reported on the train set, and on the predict set, does indicate that some degree of over–training occurred, and that the predict set results might possibly be improved by cross-validation or similar schemes. A validation set and simple cross-validation stopping criterion were used in Experiment 2; this scheme, along with a straightforward quadratic model cost ("weight-decay") (Hertz *et al.*, 1986) penalty, did produce more regularized models, decreasing the performance gap between train and predict sets.

Networks yielding new classifications of secondary structure were obtained from random initial weight values chosen from the uniform distribution between -0.2 and 0.2. However, random initial conditions suffer to a certain extent from shallow local minima. We treated this problem by repeated runs from different, random, initial weight values. One could attempt to ameliorate the problem by first separately training both the sequence and structure nets to predict the standard Kabsch and Sander classes (using conventional backpropagation), and then using these synaptic weights as the initial values of a two-network run. However, we found that the initial minimum is so deep that nothing new develops — the Kabsch and Sander definition remains unless one initializes the network with random initial weight values.

The database used in all experiments consisted of 105 proteins and is identical to that used in previous investigations (Kneller *et al.*, 1990; Stolorz *et al.*, 1992). The proteins were divided into two groups: a set of 91 "training" proteins, and a distinct "prediction" set of 14 proteins. The resulting database is similar to the database used by Qian and Sejnowski (1988) in their neural network studies of conventional secondary–structure prediction. The training and prediction sets were chosen in such a way as to contain little homology between the two sets. When comparison to predictability of conventional secondary–structure classes was needed, we defined the conventional alpha, beta and coil states using the Kabsch and Sander definitions and therefore these states are identical to those used in previous work (Kneller *et al.*, 1990; Stolorz *et al.*, 1992). A window size of 13 residues resulted in $16,028$ train set examples and 3005 predict set examples. Effects of other windows sizes have not yet been extensively tested. All results, including conventional backpropagation training of Kabsch and Sander classifications, as well as two-net training of our new secondary–structure classifications, did not employ an extra symbol denoting positions in a window that extended past the ends of a protein. Use of such a symbol could further increase accuracy.

6.1. Experiment 1

The best results have been obtained with the Mathews objective function using an architecture of five hidden units in each network. Therefore the "left–hand" (sequence window) and "right–hand" (tertiary–structure window) networks had feedforward architectures of $(260 \longrightarrow 5 \longrightarrow 1)$ and $(52 \longrightarrow 5 \longrightarrow 1)$ respectively[1]. Here, $260 = 13 \times 20$ for the unary encoding of each of 13 contiguous amino–acid residues, and $52 = 13 \times 4$ for the $\cos(\phi), \sin(\phi), \cos(\psi), \sin(\psi)$ representation for each of the same 13 residues. Adjacent layers were fully interconnected. Hidden and output units ("neurons") employed sigmoidal activation functions.

Training was accomplished with the *gradient–less* Powell minimization procedure described in (Press *et al.*, 1988) and not by conventional backpropagation that employs gradients[2]. The Mathews correlation function was designed for use on dichotomous data, and thus the most natural implementation for two networks employs binary decisions — hard classes. Hard classification introduces discontinuities in the overall objective function, making true gradient techniques impossible. Our initial tests of gradient optimization procedures for a smoothed approximation to Mathews showed that local minima were a significant problem. Local minima seemed to be much less of a problem using the gradient–less Powell procedure, although this point was not intensively investigated.

If one assigns the name "Xclass" to the newly–defined structural class, then we found that one can evolve paired networks that classify local windows of structure into a "Xclass/NotXclass" dichotomy with higher predictability than the predictability of the conventional, alpha, beta, coil secondary–structure classes. Results of the two network training using the protocol of Experiment 1 is reported in Table 1. It may be seen that the Mathews coefficient on the prediction set of the newly–defined secondary–structure classes is -0.43. This result is for a two-state dichotomy. To compare the predictability of these new two-state classes to the conventional three–state secondary–structure classes of alpha, beta and coil, it is necessary to train three backpropagation networks to perform three separate dichotomies into alpha/not-alpha, beta/not-beta and coil/not-coil. These results are also reported in Table 1. It may be seen that the predictability of the new two-state dichotomies are significantly higher than any of the alpha/not-alpha, beta/not-beta or coil/not-coil dichotomies. Adding hidden units gives negligible accuracy increase for predicting the traditional classes; however they are crucial for accurately predicting the new classes.

The negative sign of the two-network result indicates anti-correlation — a feature allowed by our objective function. The sign of the correlation is easily assessed on the train set and then can be trivially compensated for during prediction.

Table 1. Mathews correlation values (C_i) between sequence and structure network classifications, on training and prediction sets of examples. For networks trained in Experiment 1, corresponding values for Xclass classifications are compared with values for neural networks trained for dichotomous prediction of traditional classes α-helix, β-sheet, and coil.

	C_i on Train Set	C_i on Pred Set
Xclass	−0.51	−0.43
α-helix	0.37	0.33
β-sheet	0.31	0.26
Coil	0.41	0.39

A natural question to ask is whether the new classes are simply related to the more conventional classes of alpha-helix, beta-strand, and coil. A simple answer is to compute the Mathews correlation coefficient of the new secondary–structure classes with each of the three Kabsch and Sander classes, for those examples in which the sequence network agreed with the structure network's classification. The correlation with Kabsch and Sander's alpha-helix is highest: a Mathews coefficient of 0.25 was obtained on both the train set and predict set. There is therefore a significant degree of correlation with the conventional classification of alpha-helix, but significant differences exist as well. The new classes are a mixture of the conventional classes, and are not solely dominated by either alpha, beta or coil.

Conventional alpha-helices comprise roughly 25% of the data (for both train and predict sets), while the new Xclass comprises 10%. It is quite interesting that an evolution of secondary–structure classifications starting from random initial conditions, and hence completely unbiased towards the conventional classifications, results in a classification that has significant relationship to conventional helices but is more predictable from amino-acid sequence than conventional helices. In Table 2 we compare the assignment of structural features into Xclass/NotXclass categories, with the conventional assignment of structural features into alpha-helix, beta-strand and coil, for the protein *Actinidin* (which is in the predict set). The similarity, and differences, of Xclass secondary structure to conventional alpha-helices is apparent.

6.2. Experiment 2

The fastest experimental implementation of our method employed gradient–based optimization of the standard correlation, ρ, between two feed-forward networks. Discrete ("hard") classes were implemented for prediction, though the objective function was computed over real-valued network outputs, thereby ensuring smoothness. (See the Appendix.) A conjugate-gradient procedure (Press *et al.*, 1988) with a sophisticated line–search component performed the optimization. The neural networks were built, trained,

Table 2. Three representations of Actinidin (sulfhydryl proteinase) are displayed below. (Brookhaven Protein Designator: 2ACT.) The four groups of three lines illustrate the Kabsch and Sander definition of secondary structure (Helix, Beta, and Coil) in relation to the predicted Xclass secondary structure, and the target Xclass secondary structure, for protein 2ACT. Top line in each group: H=Helix, B=Beta chain, "-"=Coil, representing the conventional Kabsch and Sander secondary–structure classes. Second line in each group: "1"=Xclass, "."=NotXclass, representing the predicted (left–hand network) Xclass secondary–structure categories. Third line in each group: "1"=Xclass, "."=NotXclass, representing the target (right–hand network) Xclass secondary–structure classes.

```
----------------HHHHHHHHHHHHHHHHHH-------HHHHHHH--
...1.1...........111..11.1..................11...
................111111111111111111........11111.

-----------HHHHHHHHHHH------------------HHHHH----
......................................111111.1..
1.........111.11111111....................1.11..

--BBBB-----HHHHHHHHHH--BBBBB----HHHHH---BB--------
............11111.........1111111111............
.............11111111...........111111..........

-BBBBBBBBBBB--BBBBBBB-----------BBBBB---------------
................................1111............
................................................
```

and graphically displayed with the aid of the Xerion neural network simulation software (Connectionist Research Group, 1990).

The same basic network architectures as in Experiment 1 were used, with varying numbers of hidden units. The specific predictive accuracy results quoted below were produced by networks with two hidden units each. A somewhat unique cross-validation method was used to determine when to stop training. Because the Mathews correlation measure was to be used to assess the success of training and prediction, even though ρ-correlation was used as the objective function, training was terminated when the *Mathews* correlation between networks, on the validation set, began to decrease. The validation set comprised 1000 examples chosen randomly and removed from the training set.

The experiments with this implementation proceeded much faster than the Experiment 1 simulations. On a Silicon Graphics Iris 4D machine, slower and significantly less parallel than the CM5 system, it was possible to train each network to acceptable predictability levels within an hour or two at most.

This speedup allowed us to perform a larger number of neural network simulations and in fact to explore in some depth a particular region of classification space. Of 75 training runs from different random initial weight configurations with the same $(260 \longrightarrow 2 \longrightarrow 1)$ and $(52 \longrightarrow 2 \longrightarrow 1)$ network architectures, 9 resulted in prediction set correlations $|C_i| \geq 0.39$. Of these nine well-trained systems, four were very similar, in terms of producing classifications very highly correlated ($|C_i| > 0.8$) with each other. Two of these, which we call Y1class and Y2class, had the highest predictability values in Experiment 2 and also display closer relationships with two of the traditional Kabsch and Sander classes than does Xclass. Closest to the traditional classes is Y3class, which is also slightly less predictable than the other Yclass classifications. These results are summarized in Tables 3 and 4.

As the tables indicate, the Y1class and Y2class classifications are more predictable than the standard secondary–structure classes, with Mathews values of -0.42 and 0.42 as compared to the maximum of 0.39 for coil. These prediction set correlation values are almost equal to the −0.43 computed for Xclass, though the training set values are not nearly in the Xclass range.

Table 3. Mathews correlation values (C_i) between sequence and structure network classifications, on training and prediction sets of examples. For networks trained in Experiment 2, corresponding values for Y1class, Y2class, and Y3class classifications are compared with values for neural networks trained for dichotomous prediction of traditional classes α-helix, β-sheet, and coil.

	C_i on Train Set	C_i on Pred Set
Y1class	−0.46	−0.42
Y2class	0.44	0.43
Y3class	0.43	0.39
α-helix	0.37	0.33
β-sheet	0.31	0.26
Coil	0.41	0.39

Table 4. Mathews correlation values, for trained network on prediction set of examples, for each of Y1class, Y2class, and Y3class as measured against each other and against traditional classes α-helix, β-sheet, and coil.

	Y1class	Y2class	Y3class	α-helix	β-sheet	coil
Y1class	1.00	−0.89	−0.82	−0.32	0.08	0.37
Y2class		1.00	0.84	0.32	−0.00	−0.30
Y3class			1.00	0.46	−0.01	−0.34

7. Conclusions and Future Work

A primary goal of this investigation is to evolve highly predictable protein secondary–structure classes. Ultimately, such classes could be used, for example, to provide constraints on tertiary–structure calculations. The results described above come from preliminary investigations, and our goal will be to improve accuracy still further. However, it is now clear that the use of two, co-evolving, adaptive networks defines a novel machine learning paradigm that has allowed us to evolve new definitions of secondary structure that are significantly more predictable from primary amino–acid sequence than the conventional definitions.

Our ongoing efforts are aimed towards developing a better understanding of the structure classes found above, as well as towards developing new, computationally efficient techniques for evolving predictable classes of protein secondary structure. Effects of different neural architectures, window sizes, amino-acid representations, as well as use of non-neural algorithms to replace the two adaptive networks will be further investigated. Structure classifications with three or more classes will be evolved. The concept that agreement between the two adaptive networks results from compatible clusterings between the amino-acid representation and the structural representation, will be further developed into cluster-based, non-neural algorithms and Bayesian joint–space density modeling methods. It is essential to develop methods to overcome the local minima problem, which is particularly prevalent for the most useful objective function, mutual information.

The classifications produced by our methods are presently more predictable from amino-acid sequences than are traditional secondary–structure classifications, as measured by the Mathews correlation function. The constraint of predictability, i.e., of agreement between the outputs of the sequence network and the structure network, was explicitly built in via the objective function. What other useful properties should a secondary–structure classification have, and how may these be implemented in an objective function? For example, to effectively use secondary–structure predictions as an aid to tertiary–structure prediction, the secondary–structure classification must significantly constrain the $\Phi\Psi$ angles. It is an open problem to define, and maximize, an objective function that quantifies the structural constraints induced by a secondary structure classification. However, towards this end, we believe that the same Bayesian/MDL framework which provides the basis for modeling a single space (Rissanen, 1986; Zemel, 1994) can be used to advantage in managing the within–space *versus* between–spaces tradeoffs inherent in joint–space modeling. We are exploring this idea.

Finally, we note that the methods described here might be usefully applied to other cognitive/perceptual or engineering tasks in which correlation of two or more different representations of the same data is required (de Sa, 1994).

Acknowledgments

We gratefully acknowledge useful discussions with Geoff Hinton, Peter Dayan, Sue Becker, and Joe Bryngelson. Sue Becker's contribution of software that was used in the early stages of this project is much appreciated. We also thank the reviewers and the editor for insightful comments and helpful suggestions. The research of Alan Lapedes and Robert Farber was supported in part by the U.S. Department of Energy. The authors would like to acknowledge the hospitality of the Santa Fe Institute, where much of this work was performed.

The standard product-moment correlation and its derivative are:

$$\rho(y_1, y_2) = \sum_p (y_1^{(p)} - \overline{y_1})(y_2^{(p)} - \overline{y_2})$$

$$\frac{\partial \rho(y_1, y_2)}{\partial y_1^{(p)}} = \frac{1}{V_1 V_2}(y_2^{(p)} - \bar{y_2}) - \frac{V_2}{V_1} V_{12} \frac{1}{(V_1 V_2)^2}(y_1^{(p)} - \bar{y_1})$$

where $V_i = \sum_p (y_i^{(p)})^2 - (\sum_p y_i^{(p)})^2$ and $V_{12} = (\sum_p y_1^{(p)} y_2^{(p)}) - (\sum_p y_1^{(p)})(\sum_p y_2^{(p)})$.

Sigmoidal output functions, weight–cost terms, and their derivatives may be found in any neural network reference, such as (Hertz *et al.*, 1986).

The training of the right–hand network is entirely analogous. The two networks are trained simultaneously, step–for–step, in our current implementations. Other objective functions might best be optimized by alternating–minimization procedures.

Notes

1. Each of these networks trained in Experiment 1 actually employed 2 output units. The two outputs in each network were independent, and defined two independent subnetworks, in the sense of feedforward processing and prediction. As for learning, the first output of the sequence network was trained towards correlation with the first output of the structure network, and likewise for the second output units of the respective networks. The weights on the connections between the input and hidden layers were responsible to the optimization of both sets of correlations. Interestingly, this architecture produced *better* learning and prediction results, for at least one of the two sets of corresponding outputs, than the simpler single–output case tried in other experiments.
2. The Powell algorithm works by performing successive line searches in conjugate directions, and does not require a gradient. It is generally slower, however, than gradient methods, and requires $O(N^2)$ storage, as compared with $O(N)$ for conjugate–gradient methods.

References

Abola, E. E., Bernstein, F. C., Bryant, S. H., Koetzle, T. F., & Weng, J. (1987). Protein data bank. In *Crystallographic databases*. International Union of Crystallography.

Becker, S. & Hinton, G. (1992). A self-organizing neural network that discovers surfaces in random-dot stereograms. *Nature*, 355, 161–163.

Becker, H. S. (1992). *An Information-theoretic Unsupervised Learning Algorithm for Neural Networks*. Ph.D. thesis, Department of Computer Science, University of Toronto.

Chou, P. Y. & Fasman, G. D. (1978). Prediction of the secondary structure of proteins from their amino acid sequence. *Advances in Enzymology*, 47, 45–147.

de Sa, V. R. (1994). Learning classification with unlabeled data. In *Advances in Neural Information Processing Systems 6*, San Francisco. Morgan Kaufmann.

Delorme, M.-O. & Henaut, A. (1988). Merging of distance matrices and classification by dynamic clustering. *Computer Applications in the Biosciences*, 4, 453–458.

Efron, B. & Tibshirani, R. (1991). Statistical data analysis in the computer age. *Science*, 253, 390–395.

Appendix

Prediction of structural class using sequence network:

Two–class secondary–structure prediction on the pth example is defined for the (left–hand) sequence network in terms of the network output O_L on example p, the mean output over all training examples, and the post–training correlation between the outputs of the two networks, as follows.

```
Run example $p$ through network;
*if* $(O_L^{(p)} < \bar{O}_L)$ *then* class := $CLASS1$
*else* class := $CLASS2$;
*if* $(\rho(O_L, O_R) \geq 0)$ *then*
    *if* (class = $CLASS1$) *then* predict := $CLASS1$
    *else* predict := $CLASS2$;
*else*
    *if* (class = $CLASS1$) *then* predict := $CLASS2$
    *else* predict := $CLASS1$;
*return* predict;
```

Essentially, this says that if a network's output is below average, then the corresponding input is in one class; if it is above average, then the input is in the other class. Then, in making the prediction of the other network's output, use the first network's classification as the prediction, unless the correlation between the networks is negative, in which case predict the opposite.

This is easily extended to handle $k > 2$ classes or different correlation or "agreement" measures.

Training of two–net system using ρ-correlation objective function:

E is the objective function, which in this case is $E = -\rho^2$. O_L is, as above, the output of the left–hand network. w is a synaptic weight in the network. We wish to train the network by adapting the weights in order to minimize the value of the objective function, over all of the training examples p.

Training is accomplished by a conjugate-gradient optimization algorithm (Press *et al.*, 1988), where the gradients are computed as follows.

$$\frac{\partial E}{\partial w} = \frac{\partial E}{\partial \rho} \frac{\partial \rho}{\partial O_L^{(p)}} \frac{\partial O_L^{(p)}}{\partial w}$$

$$\frac{\partial E}{\partial \rho} = -2\rho$$

Farber, R., Lapedes, A., & Sirotkin, K. (1992). Determination of eukaryotic protein coding regions using neural networks and information theory. *Journal of Molecular Biology*, 226, 471–482.

Fitch, W.M. (1981). A non-sequential method for constructing trees and hierarchical classifications. *Journal of Molecular Evolution*, 18, 30–37.

Connectionist Research Group. (1990). *Xerion Neural Network Simulator Libraries and Manual Pages; version 3.183*. Department of Computer Science, University of Toronto.

Hertz, J., Krogh, A., & Palmer, R. (1986). *Introduction to the Theory of Neural Computation*. Menlo Park, CA. Addison-Wesley (Santa Fe Institute Studies in the Sciences of Complexity).

Hunter, L. & States, D. (1992). Bayesian classification of protein structure. *IEEE Expert*, 7(4), 67–75.

Hunter, L., Harris, N., & States, D. (1992). Efficient classification of massive unsegmented datastreams. In *Proceedings of the Ninth International Conference on Machine Learning*, San Mateo, CA. Morgan Kaufmann Associates.

Holland, J., Holyoak, K., Nisbett, R., & Thagard, P. (1986). *Induction: Process of Inference, Learning and Discovery*. Cambridge, MA. MIT Press.

Kabsch, W. & Sander, C. (1983). Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, 22, 2577–2637.

Kneller, D. G., Cohen, F. E., & Langridge, R. (1990). Improvements in protein secondary structure prediction by an enhanced neural network. *Journal of Molecular Biology*, 214, 171–182.

Korber, B. T. M., Farber, R. M., Wolpert, D. H., & Lapedes, A. S. (1993). Covariation of mutations in the V3 loop of HIV-1: An information-theoretic analysis. *Proceedings of the National Academy of Sciences, USA*, 90, 7176–7180.

Lapedes, A., Barnes, C., Burks, C., Farber, R., & Sirotkin, K. (1990). Application of neural networks and other machine learning algorithms to DNA sequence analysis. In G. I. Bell and T. G. Marr (Eds.), *Computers and DNA*. Menlo Park, CA. Addison-Wesley (Santa Fe Institute Studies in the Science of Complexity.)

Lapedes, A. S., Steeg, E. W., & Farber, R. M. (1994). Neural network definitions of highly predictable protein secondary structure classes. In *Advances in Neural Information Processing Systems 6*, San Francisco. Morgan Kaufmann.

Maclin, R. & Shavlik, J. W. (1992). Using knowledge-based neural networks to improve algorithms: Refining the Chou-Fasman algorithm for protein folding. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, San Francisco. Morgan Kauffman.

Mardia, K. V., Kent, J. T., & Bibby, J. M. (1979). *Multivariate Analysis*. New York, Academic Press.

Michalewicz, Z. (1986). *Genetic Algorithms*. Menlo Park, CA. Addison–Wesley (Santa Fe Institute Studies in the Sciences of Complexity).

Pauling, L., Corey, R., & Branson, H. R. (1951). The structure of proteins: two hydrogen-bonded helical configurations of the polypeptide chain. *Proceedings of the National Academy of Sciences, USA*, 37, 205–211.

Perutz, M. F. (1951). New x-ray evidence on the configuration of polypeptide chains; polypeptide chains in poly-γ-benzyl-L-glutamate, keratin, haemoglobin. *Nature*, 167, 1053–1059.

Press, W. H., Flannery, B. P., Teukolsky, S. A., & Vetterling, W. T. (1988). *Numerical Recipes in C*. London, Cambridge University Press.

Prestrelski, S. J., Williams, A. L. Jr., & Liebman, M. J. (1992). Classification of protein secondary structure. I. Overview of the methods and results. *Proteins: Structure, Function, and Genetics*, 14, 430–439.

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

Rissanen, J. (1986). Stochastic complexity and modeling. *Annals of Statistics*, 14 (3), 1080–1094.

Rumelhart, D. & McClelland, J. (1986). *Parallel Distributed Processing*. Boston. MIT Press.

Schmidhuber, J. (1992). Discovering predictable classifications. Technical Report CU-CS-626-92, Department of Computer Science, University of Colorado.

Schulz, G. E. & Schirmer, R. H. (1979). Prediction of secondary structure from the amino acid sequence. In *Principles of Protein Structure*. New York. Springer-Verlag.

Skolnick, J & Kolinski, A. (1991). Dynamic Monte Carlo simulations of a new lattice model of globular protein folding, structure and dynamics. *Journal of Molecular Biology*, 223, 583–597.

Stolorz, P., Lapedes, A., & Yuan, X. (1992). Predicting protein secondary structure using neural net and statistical methods. *Journal of Molecular Biology*, 225, 363–378.

Unger, R., Harel, D., Wherland, S., & Sussman, J. L. (1989). A 3D building blocks approach to analyzing and predicting structure of proteins. *Proteins: Structure, Function, and Genetics*, 5, 355–363.

Zemel, R. (1994). *A Minimum Description Length Framework for Unsupervised Learning*. Ph.D. thesis, Department of Computer Science, University of Toronto.

Zhang, X. & Waltz, D. (1993). Developing hierarchical representations for protein structures: An incremental approach. In L. Hunter (Ed.), *Artificial Intelligence and Molecular Biology* (pp.195–209). Menlo Park, CA, AAAI Press (MIT Press).

Received October 21, 1993
Accepted December 6, 1994
Final Manuscript January 9, 1995