Design of an Auto-associative Neural Network with Hidden Layer Activations that were used to Reclassify Local Protein Structures

Xiru Zhang, Jacquelyn S. Fetrow, and George Berg

Departments of Biological Sciences and Computer Science, University at Albany, SUNY, Albany, NY

I. Introduction

The prediction of three-dimensional protein structure from the primary amino acid sequence is a fundamental unsolved problem in molecular biology (1). Success in solving this problem depends on finding the "rules" that govern the correspondence between amino acid sequences and the structures they form. One potential solution is to identify similar substructures in different proteins, examine their amino acid sequences for common patterns, then using these patterns to predict the substructure classification of a given amino acid sequence.

This approach is most commonly done using the regular secondary structures α-helix and β-strand (2). Their repeating backbone dihedral angles and regular hydrogen bonding patterns make them easily recognizable by the human eye in crystal structures of globular proteins. The use of helices and strands to classify and predict protein structure has certain limitations. Several different algorithms have been developed to automatically determine the location of the helices and strands in proteins of known structure (3, 4). Although these different techniques usually agree on the general location of a helix or strand, they disagree on the exact position of the ends of the secondary structure; furthermore, the various algorithms often completely disagree on the location of non-typical helices and strands that are fairly common in proteins (Fetrow and Berg, unpublished results).

A second drawback of secondary structure as a classification scheme is that the categories lack specificity. Generally, all structure that is not helix or strand is lumped into the default category "coil" which accounts for about 50% of the secondary structure of globular proteins. Observation of these structures demonstrates that they are not random and should not be lumped into a single category (2). Thus, even if the coil regions could be predicted accurately, it
different sets of protein structure data were used to train the network, (2) they have interesting relations to the traditional secondary structures of helix and strand, and (3) there are strong amino acid preferences for many of the categories, suggesting that prediction of these structures from primary amino acid sequences may be possible (17; Zhang, Berg, and Fetrow, manuscript in preparation).

II. Methods

A. Protein Database

The protein database consisted of 75 sequence chains from 74 globular proteins with structures obtained from the Brookhaven database (18). These structures were selected to have a resolution of 2.5 Å or better and an R value of 0.3 or less. No protein had more than 50% sequence identity to any other protein. The database was divided into two disjoint groups, Data Set 1 and Data Set 2 to train the networks. The database is described in more detail in reference 17.

B. Encoding of the Structure of the Protein Segments

The alpha carbon geometry of continuous seven-residue segments from the protein database was used as input into the neural network. The database contained 12,664 of these segments. The geometry of these segments was defined by three parameters: 1) the distances between non-neighboring alpha carbons, 2) virtual bond angles between each three consecutive alpha carbons, and 3) the virtual dihedral angle between each four consecutive alpha carbons (Figure 1).

These parameters were encoded for input to a neural network as follows. The distance distribution in many seven-residue segments is bimodal; thus, each of the fifteen distances was encoded using two input units to represent the two peaks of the distribution. If the distance fell within the first peak, the first unit was set to a value representing the portion of the distance within the first peak, normalized to the value range [0,1], and the second unit was set to zero. If the distance fell within the second peak of the bimodal distribution, the first unit was set to 1, and the second unit represented the difference between the cutoff point between the two peaks and the distance, normalized to the range [0,1]. Using this scheme 30 units represent the 15 distances in each seven-residue segment. Each of the five virtual bond angles between alpha carbons, with a value between 0 and 180°, was encoded by one input unit. The value of these units was also normalized to the range [0,1]. Each virtual dihedral angle was represented by two input units, one each for the sine and cosine of the angle, normalized to the value range [0,1]; thus, eight input units encode the four dihedral angles. In total, 43 units represented the structure of a given seven-residue segment. These 43 units are called the residue-feature-vector for each seven-residue segment.
C. Auto-associative Artificial Neural Network

Most artificial neural network models are trained to perform a specific task. This is done by presenting a series of input patterns to the network. For each pattern, the network calculates an output pattern that is compared to the desired output pattern for this input. In order to reduce the difference between actual and desired outputs, the difference between the two is measured and the state of the network is changed by a learning heuristic, frequently error backpropagation (15), in order to reduce the difference. When the learning is successful, the network’s behavior changes incrementally, and after many presentations of the data, the actual outputs correspond closely to the desired outputs. A successful network’s training terminates when it is able to correctly recreate the desired outputs within some tolerance parameter. The input is presented as a vector of input units, with values usually in the range [0,1]. After the input units, there is another vector of hidden units. Every hidden unit and input unit are associated by a weight between them, that takes on a real-numbered value, initially set to a random value drawn from the uniform distribution in [-1,1]. After the hidden layer, there is a layer of output units, and weights connecting the hidden and output units.

When a pattern is presented to the network as an input unit value vector, these values and the weights are used to calculate activation values for the hidden units for this pattern. For each hidden unit, \( j \), its net input, \( \text{net}_j \), is

\[
\text{net}_j = \sum_i w_{ij} a_i
\]

where \( a_i \) is the activation value of input unit \( i \), and \( w_{ij} \) is the weight connecting input unit \( i \) and hidden unit \( j \). The net input is then filtered through a squashing function, \( f \), to scale it into the range [0,1], producing the activation value, \( a_j \), for the hidden unit \( j \)

\[
a_j = f(\text{net}_j) = \frac{1}{1 + e^{-\text{net}_j + \Theta_j}}
\]

where \( \Theta_j \) is the bias term of unit \( j \). The bias term can be thought of as a weight from a special unit with a permanent activation value of 1. Once the hidden unit activations are calculated, they are used along with the weights between the hidden and output units to calculate the activation values of the output units. These calculations are performed just as those described for the hidden units.

Each output unit's activation value is then compared to the desired value the unit should produce for the current pattern. The difference between the two values is used to calculate the unit's error. For input pattern, \( p \), and output unit, \( k \), the error, \( \delta_{pk} \), is calculated to be

\[
\delta_{pk} = f'(\text{net}_{pk})(t_{pk} - a_{pk})
\]

where \( f' \) is the derivative of the squashing function \( f \), \( \text{net}_{pk} \) is the net input to unit \( k \) for pattern \( p \), \( t_{pk} \) is the desired value for output unit \( k \) for pattern \( p \), and \( a_{pk} \)

\[
\delta_{pk} = f'(\text{net}_{pk})(t_{pk} - a_{pk})
\]
units. The residue-feature-vectors were used as both input and output patterns for this network, called GENE-REP (GENERator of REPRESENTations) (17). Using the representation for the residue-feature-vectors described above, both the input and output layers of the network used 43 units. The hidden layer for this network used eight units (Figure 2).

The networks were trained for approximately 1500 epochs on a Connection Machine CM-5 until the RMS error between the input and output patterns was at most 0.01. The learning rate, \( \eta \), was initially set to 0.000001 and gradually increased to 0.005 during the training. The momentum constant, \( \alpha \), was 0.9. After training, the network was run one additional epoch, without learning, on the residue-feature-vector patterns and the values of the hidden layer units (the residue-state-vectors) were recorded for each seven-residue segment.

D. K-means Clustering Algorithm

The residue-state-vectors produced by the network were then grouped using a k-means clustering algorithm (16). To classify \( N \) numeric vectors into \( n \) categories:

1. Randomly select \( n \) vectors as “seeds” for the \( n \) clusters.
2. For each of the remaining vectors, find the closest seed and assign the vector to its cluster.
3. Compute the center of each cluster.
4. Using the centers as new seeds, repeat steps 2-3 until the percentage of vectors clustered differently in successive iterations falls below \( p \).
5. Compute the deviation within each cluster, and the mean deviation of all of the clusters.
6. Repeat steps 1-5 \( m \) times and select the clustering that has the lowest mean deviation.

The residue-state-vectors were clustered using cluster sizes, \( n \), of three through ten. For each of these clusterings, an iteration parameter, \( m \), of 40 was used. The percentage threshold, \( p \), was 0.1%. The cluster assignments of the residue-state-vectors from the clustering algorithm are the Structural Building Blocks (SBBs) and are the categories of local structure that form the basis for this study.

III. Results and Discussion

It has been shown that the hidden unit values of a properly trained auto-associative neural network represent the important features of the network’s inputs (15). In addition, if the hidden unit values are used as the canonical (encoded) representation of the inputs, the input layer to hidden layer portion of the trained network can be used as an encoder for the representations, and the hidden layer to output layer portion as a decoder.

An auto-associative network should have the smallest hidden layer that still allows the network to learn the auto-association task. This forces the hidden unit values vector to become a concise representation of the information in the inputs.

In order to learn the task with a small number of hidden units, the units tend to represent distinct features of the input patterns. And since their activations are calculated using the squashing function, each has the value range \((0,1)\). Such properties make the residue-state-vectors useful for analyses such as clustering, because clustering algorithms often assume the input data to have these properties for their results to be reliable.

The use of the residue-state-vectors is a major difference between this work and other efforts to reclassify protein structures: we separate the issue of representation from that of classification. Our measure of similarity among local structures is based on the canonical encoding in the residue-state-vectors constructed by the auto-associative neural network, rather than simply using “raw data” of a single type of measure, such as alpha carbon positions or \( \phi \) and \( \psi \) angles, neither of which are perfect similarity measures.

One auto-associative network was trained on each of two disjoint sets from the protein database, Data Set 1 and one on Data Set 2. After training, the residue-state-vectors for the proteins in Data Set 2 were separately calculated using each network. Comparisons of the separate categorizations from the two different networks showed that the training was general and not specific to the individual data set on which the network was trained. Analysis of these separate categorizations indicates that clustering into six SBB categories gives the most general, reproducible categories (17, Zhang, Berg, and Fetrow, manuscript in preparation).

Analysis of the SBB categories provides interesting results. Though no information about helix or strand was provided as input to the networks, the system automatically developed categories for \( \alpha \)-helix and \( \beta \)-strand. The network also seems to uniquely recognize the N-terminal caps and C-terminal caps of helices, as well as analogous caps for strand regions (17; Zhang, Berg, and Fetrow, manuscript in preparation). Other structurally significant aspects of the SBB categories are being studied. There are strong preferences of some amino acids for certain SBB categories; thus, the SBB categories for the regions of proteins that have been less studied, the “coil” regions, may also be meaningful and predictable. We are currently looking into these issues.

IV. Conclusions

A major goal of this work was to develop “secondary-structure-like” local structural categories for every part of a globular protein. These will be used in the ongoing effort to predict the tertiary structure of globular proteins from their amino acid sequence. We have developed and described here an auto-associative neural network that is capable of reducing the large amount of information necessary to describe the geometry of a segment of protein into vectors of length eight. The vectors for all protein segments can then be clustered to yield novel secondary structure classifications or Structural Building Blocks (SBBs). SBBs have interesting structural properties and often have strong preferences for certain types of amino acids. This offers hope that these categorizations can be the basis of improved prediction of protein tertiary structure from sequence data.
Acknowledgments

JSF and GB are supported by NSF grant BIR-9211256. The authors wish to thank David Waltz for his invaluable contributions during the initial stages of this work.

References


I. INTRODUCTION

Amino-acid mutations in proteins are accepted at the level of the organism by natural selection acting on advantageous mutations or by random drift where mutations are selectively neutral, or at least non-lethal, with respect to the function or survival of the organism. Many amino-acid mutations produce detrimental effects and cause genetically associated diseases, yet are still accepted during evolution. The acceptability of a mutation in a protein depends largely on whether the amino-acid substitution is compatible with structural features of the local environment. Overington et al. (1990, 1992) considered various structural features that characterise the local environment, including side-chain solvent accessibility, main-chain conformation and side-chain hydrogen bonds. All of these structural features restrain the amino-acid substitutions that are acceptable at any particular site; for example, solvent inaccessible polar residues, such as aspartic acid, serine, threonine and glutamine, which hydrogen bond to main-chain amides, are highly conserved or invariant within protein folds (Overington et al., 1990, 1992).

The tertiary structures of protein families are much more conserved than their primary structures (Greer, 1981; Bajaj and Blundell, 1984; Chothia and Lesk, 1986; Johnson et al., 1990a, b), such that the arrangement of helices, strands and thus the main-chain polar groups (carbonyls and amides) remains fairly constant. Even the packing arrangements of main-chain amides and carbonyls around amino-acid side-chains are not random (Singh and Thornton, 1990), and these backbone groups may influence amino-acid substitutions through either steric or electrostatic effects. For these reasons, we have studied the effects of interacting main-chain polar groups on the pattern of amino-acid substitutions in families of homologous proteins. Here, we show how combinations of main-chain amide and carbonyl groups in the local environment of amino-acid residues bias amino-acid replacements at given positions of the protein fold.