Felix Amoruwa, 15448235

Isaac Noble

Computational Engineering Science

Using Artificial Neural Networks to Predict Secondary Structure of Proteins

Bioengineering 131 – Introduction to Computational Molecular Biology

12/16/04

For the final project, we created an artificial neural network that attempts to predict the secondary sequence structure of proteins.

INTRODUCTION:

Our final project performs a secondary structure prediction on primary protein sequences by using an artificial neural network model that implements a learning algorithm. Using the primary protein sequences and the corresponding secondary structures sequence as input, upon training an artificial neural network (ANN) with approximately 42 different proteins, our ANN should be able to predict the secondary sequence of any given primary protein sequence with above 50% accuracy.

The underlying purpose of devising a computing model stems from the fact that there are millions of proteins spanning the 7 kingdoms on our planet; only a fraction of these proteins that have been discovered have been successfully studied with their known corresponding secondary, tertiary, and quaternary structures. With the assistance of today's technological computing power, using computer programs to accurately model and predict the secondary, tertiary, or quaternary structures of any given primary protein sequence, biologists can readily have secondary sequences for further analysis of three-dimensional protein structure. Furthermore, biologists can then conduct further experiments and analyses based on the computer representation of these protein sequences to classify and determine the domains of proteins. With this knowledge readily available, it becomes much easier for the biologist to predict the biological role of proteins and the mechanism through which they perform their respective functions.

The artificial neural network used, devised by Dr. Steven Noble, incorporates simple processing elements with a high degree of interconnectedness that stores "weight" values into each neuron within the network. Several layers of neurons within the network are used: input neurons, hidden layer neurons, and output neurons. Iterating through the network allows for the neural network to learn its target output through interaction with every other neuron in the network.

In order for the neural network to successfully accept the learning data and output the targeted data, the network must propagate throughout the layers of the network beginning with the input neurons, through the hidden layer(s), and to the output neurons. A predefined function that assigns a weight value to each neuron updates the weight associated with each neuron by a method called forward propagation.

The artificial neural network also incorporates a backward propagation algorithm to make sure that the “neural network” is actually learning, by updating the weights of the neurons to what the output neurons are. By starting at the output neurons, the weights connecting the output to the next layer (hidden layer) are adjusted by the backward-propagation function; this algorithm continues to do so until it reaches the input neuron layer.

Currently, there are several different methods, other than using artificial neural networks, used to predict secondary and tertiary structures of proteins that follow different approaches generating varying accuracies. Using laws of probability in conjunctions with single-residue statistics, biologists have been able to generate structure predictions. Another method of structure prediction used by biologists is the nearest-neighbors method, which looks at “n” number of neighboring amino acids from the reference amino acid, and from there, calculates probabilities to make assumptions and predictions of the structure relating to the amino acid relative to its neighbors.

MATERIALS AND METHODS:

The training data that was used on the artificial neural network comprised of forty-two different protein sequences retrieved from the RCSB (Protein Data Bank – http://www.rcsb.org/pdb/). Originally in the PDB format, a parser program, written in java, was used to manipulate each protein sequence into a format in which the neural network could accept as input. With the neural networks taking the training data as input - the primary sequence of amino acids and the corresponding secondary sequence structure representation - the neural networks could thus perform its learning methodology on the training data.

The parser incorporates three hashmaps (hashtables) that relate to the three different properties used to represent each amino acid in the primary sequence: helix propensity, beta sheet propensity, and hydrophobicity. For each property, a point system ranging between negative one and positive one was assigned to each amino acid to distinguish them. This method created the input training data file into a numerical format to be used as input in the neural network.

For the secondary structure sequence corresponding to each protein, there are eight different classes that represent the particular amino acid's secondary structure. The eight different class representations used for the secondary structure sequence are as follows: H – helix, G – 310 helix, I – pi helix, S – bend, T – turn, E – beta sheet, B – beta bridge, N – nothing. Each class was given a scoring scheme ranging from –0.9 to 0.9 for the eight output neurons. From these eight classifications, the parser also grouped similar categories together (helices, beta sheets, turns, and nothing) and assigned a unique four-number scoring scheme to represent each category for four output neurons. From here, the parser generated the training output data from which the neural network would be learning from, all in numerical format. With the training data (input and output) files in the correct format to be accepted as input into the neural network, the learning process of iterating through the neural network commenced.

A total of six artificial neural networks were used with varying parameters. For the neural networks with eight output neurons, the output neurons correspond to the eight different secondary structure class representations outlined above using an amino acid window size of eleven residues. For the neural networks with three output neurons, the output neurons correspond to the three categories of secondary structure class representation: helix, beta sheet, and turn (curl); with an amino acid window size of seventeen residues. The underlying reason behind using neural networks with varying parameters was to eventually find the optimal neural network with corresponding parameters that would generate the optimum accuracy and smallest relative error.

After using the parser to generate input files that contained primary and secondary sequence data from the forty-two proteins chosen, the converted sequences were compiled into two input files for the neural networks: training_input and training_output. The neural network also made use of a randomization function that randomly selected vector sequences from the input files to evenly distribute the learning data.

The two algorithms used in the neural networks, forward and back propagation, allow the network to eventually learn from k and m through the multilayer network. After a certain amount of iterations through the network (2 or 3 depending on which neural network was used), the neuron with the highest value after the number of iterations is the secondary structure for the middle amino acid currently being looked at.

$k =$ number of inputs

$m =$ number of neurons

$$n = k + m + 1$$
$$N_i = \begin{bmatrix} W_1 & W_2 & \dots & W_n \end{bmatrix}$$
$$S^t = \begin{bmatrix} S^t_1 & S^t_2 & \dots & S^t_n \end{bmatrix} = \begin{bmatrix} I_1 & \dots & I_k & B & V^t_1 & \dots & V^t_m \end{bmatrix}$$
$$V^{t+1}_i = sigma(N_i \bullet S^t)$$

For the back propagation, the purpose of this algorithm is to minimize the error between the output neurons and the preceding neuronal layers, eventually minimizing the error between the output and the input neurons. The error is calculated as the average squared error of the network; this error is then propagated back through the network adjust the weights of the neurons to minimize the error. This is the equivalent of performing gradient decent on the error function in an attempt to find the global minima.

$$W_i = W_i + \alpha * E^t_i * V^{t+1}_i$$
$$E^{t+1}_i = sigma'(V^{t+1}_i) * \sum_{j=1}^{m} E^t_j * W_j$$
$$E^0_i = \begin{cases} (T_i - V^0_i) * sigma'(V^1_i) \rightarrow i <= numOutputs \\ 0 \rightarrow i > numOutputs \end{cases}$$

Taking the derivative of the error with respect to the weight and then applying the chain rule derived these equations.

$$E^0 = \frac{1}{2}\sum_{i=0}^{m}\left(T_i - V_i^0\right)^2$$

$$\frac{\partial E^0}{\partial W_i} = -\left(T_i - V_i^0\right)*\frac{\partial V_i^0}{\partial W_i} = -\left(T_i - V_i^0\right)*sigma'\left(N_i \bullet S^1\right)*\frac{\partial\left(N_i \bullet S^1\right)}{\partial W_i} = -\left(T_i - V_i^0\right)*sigma'\left(V_i^1\right)*V_i^1$$

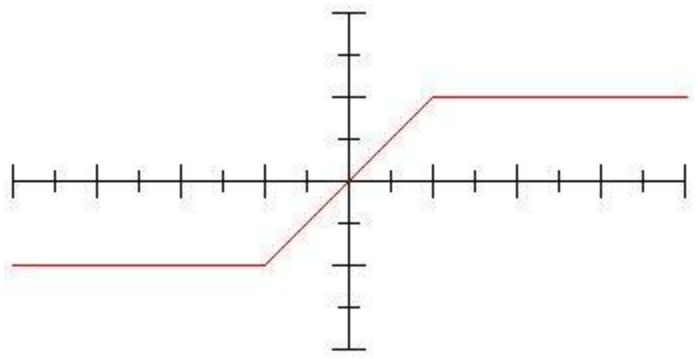
$$E^t = \frac{1}{2}\sum_{j}^{m}\left(E_j^{t-1}\right)^2$$

$$\frac{\partial E^t}{\partial W_i} = -\sum_{j}^{m}\left(E_j^{t-1}*\frac{\partial V_j^{t-1}}{\partial W_i}\right) = -\sum_{j}^{m}\left(E_j^{t-1}*W_j\right)*\frac{\partial V_i^t}{\partial W_i} = -\sum_{j}^{m}\left(E_j^{t-1}\right)*sigma'(N_i * S^{t+1})*\frac{\partial(N_i * S^{t+1})}{\partial W_i} = -\sum_{j}^{m}\left(E_j^{t-1}*W_j\right)*sigma'(V_i^{t+1})*V_i^{t+1}$$

The networks used in this project incorporated a linear sigmoid function:

$$sigma(v) = \begin{cases} -1 \rightarrow v \le -1 \\ v \rightarrow -1 < v < 1 \\ 1 \rightarrow v \ge 1 \end{cases}$$

$$sigma'(v) = \begin{cases} 0 \rightarrow v \le -1 \\ 1 \rightarrow -1 < v < 1 \\ 0 \rightarrow v \ge 1 \end{cases}$$



This network was tested on a number of test sets, the first being the logical statements, AND, OR and XOR. Interestingly enough, the neural networks are able to learn XOR with only two neurons by using two steps forward. The networks were also tested for hand written number recognition; on this data it preformed as well as a standard neural network of similar complexity.

RESULTS:

Although several neural networks were created with varying numbers of output neurons, after running multiple simulations with these neural networks on the test data, we came to conclude that the best predictions made were those using the neural networks with three output neurons, corresponding to beta sheets, helices, and curls. The number of predictions for the eight categories of secondary structures was too specific for our network to accurately learn, thus, the reasoning behind changing the output neurons in the neural network from eight to three outputs.

Our results were not nearly as good as we had hoped for; the neural network with eight output neurons was able to achieve prediction accuracy of 40% while the neural network of predicting just helix and beta sheets had a best prediction accuracy of approximately 60%. Although other secondary structure prediction programs are able to accurately predict in the seventy-percentile region, we suspect that homologs were used within their data.

Topology Results

Neurons	Inputs	Outputs	Iterations	Accuracy	Comments
33	33	8	2	38%	
33	33	8	3	39%	Most accurate overall network
66	33	8	2	38%	Experienced over training
7	33	3	2	52%	
7	51	3	2	57%	Most accurate 2^0 network
28	51	3	2	49%	Experienced over training

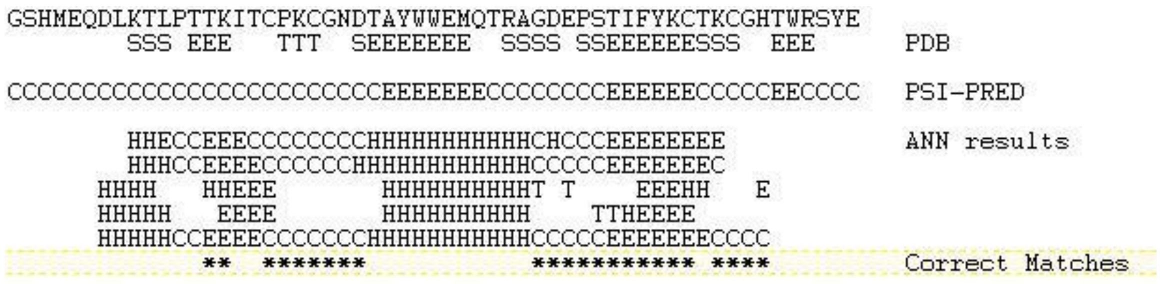
These results were obtained from a test set of forty-two proteins retrieved from various sources, that were spliced up into 10000+ samples; this included overlap which may partly explain the below average performance of the network. The network was then tested on a set of 1000 samples that it was not trained on. The accuracy given is in terms of Q3 and Q8:

$$Q3 = \frac{P_H + P_E + P_C}{Total} \qquad Q8 = \frac{P_H + P_G + P_E + P_B + P_S + P_T + P_I + P_{none}}{Total}$$

Where P_i=the number of correctly predicted structures, and Qi = the accuracy of the prediction, the Q gave a good idea of how well the neural network learned the trained mapping. The reason why this is a poor measure of accuracy is because fifty-percent of the amino acid residues were classified as having a curl structure, thus, the network could achieve fifty-percent accuracy by choosing curl every time.

For predicting the secondary structure sequence ab initio, another program was used though could combine multiple networks, as well as homologs to the target sequence. This allows for more accurate predictions in theory, homologs were not used to aid the prediction due to lack of time. Here are the predicted secondary structures for the two target sequences:

Target 1

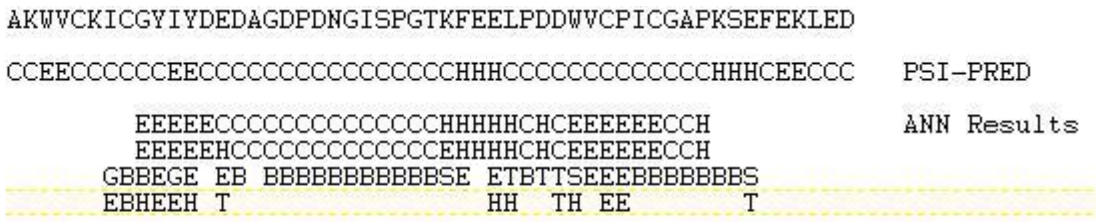


```
GSHMEQDLKTLPTTKITCPKCGNDTAYWWEMQTRAGDEPSTIFYKCTKCGHTWRSYE
        SSS EEE   TTT  SEEEEEEE  SSSS SSEEEEEEESSS  EEE      PDB

CCCCCCCCCCCCCCCCCCCCCCCCCEEEEEEECCCCCCCEEEEEECCCCCEECCCC   PSI-PRED

        HHECCEEECCCCCCCCHHHHHHHHHHHCHCCCEEEEEEEEE           ANN results
        HHHCCEEEECCCCCCHHHHHHHHHHHHHCCCCCEEEEEEEEC
      HHHH   HHEEE       HHHHHHHHHHT T    EEEHH   E
      HHHHH   EEEE       HHHHHHHHHH    TTHEEEE
      HHHHHCCEEEECCCCCCCHHHHHHHHHHHCCCCCEEEEEEEECCCC
             **  *******           *********** ****         Correct Matches
```

Target 2



```
AKWVCKICGYIYDEDAGDPDNGISPGTKFEELPDDWVCPICGAPKSEFEKLED

CCEECCCCCCEECCCCCCCCCCCCCCCCHHHCCCCCCCCCCCCCHHHCEECCC   PSI-PRED

        EEEEECCCCCCCCCCCCCHHHHHCHCEEEEEEECCH            ANN Results
        EEEEEHCCCCCCCCCCCCEHHHHCHCEEEEEEECCH
      GBBEGE EB BBBBBBBBBBBSE ETBTTSEEEBBBBBBBS
      EBHEEH T                HH  TH EE       T
```

With the first target, the neural networks had a difficult time recognizing the beta sheets and instead, found them as helices. For this target, the networks achieved about fifty percent accuracy, which is not very high. For the second target, there is nothing to compare the networks to except PSI-PRED. The networks predict most of the structures PSI-PRED predicts except for the last.

For the second target, the actual structure is unknown, but the structure predicted by PSI-PRED and JPRED are very similar to what our neural network predicted. The JPRED result predicted two beta sheets near each end of the sequence, which was close to where our neural network predicted beta sheets as well. The neural network appears to have performed accurately in comparison with the other predictor programs.

DISCUSSION:

After successfully constructing six neural networks, all with varying parameters in relation to the number of output neurons and iteration steps, upon training each neural network with the training data (forty-two proteins), the test data was passed into each neural network. From here, the accuracy of the resulting predictions of secondary sequence structures varied amongst the different neural networks. It was interesting to find how subtle changes such as adding an extra hidden layer, using more input neurons, or changing the number of iteration steps for the forward and backward propagation algorithms could drastically affect the accuracy of the predictions.

Our original plan for the project was to have the artificial neural networks perform full structure prediction; we would have liked to further develop our neural networks to successfully do so, but unfortunately, time was an issue and the information necessary to perform this prediction was not readily available during the working duration of our project.

With more time available, we would have also liked to revise the artificial neural network; particularly the mathematical algorithms involved in forward and backward propagation, to predict secondary, tertiary, and quaternary structures accurately. By running multiple simulations with training and test data on more neural networks with varying parameters - number of input/output neurons, number of hidden layer neurons, iteration steps, amino acid window sizing – an optimal neural network with these parameters could be used to predict the structure sequences with a much higher accuracy.

Another change proposed was to change the scoring schema assigned to the amino acids and to the eight different classes of secondary structures in the parser program. By grouping all of the amino acids by their respective properties (acidic, basic, etc) before assigning the scoring schema, along with grouping the secondary structures together (all helices with helices, all beta sheets with beta sheets), by assigning the scoring schema based on these groupings rather than by random, we proposed that the accuracy of the structure predictions would be better as well.

REFERENCES:

1) Dr. Steven Noble
2) Russell, Stuart. Norvig, Peter. Artificial Intelligence: A Modern Approach. 2003. Pearson Education Inc. Upper Saddle River, New Jersey.
3) Protein Secondary Structure Prediction. http://www.pasteur.fr/recherche/unites/neubiomol/secstrpr.html
4) The PSI-PRED Protein Prediction Server. http://bioninf.cs.ucl.ac.uk/psipred/psiform.html
5) The RCSB Protein Data Bank. http://www.rcsb.org/pdb/