Georgia State University
Digital Archive @ GSU

Computer Science Theses | Department of Computer Science

6-9-2006

# The Relative Importance of Input Encoding and Learning Methodology on Protein Secondary Structure Prediction

Arnshea Clayton
aclayton1@student.gsu.edu

Follow this and additional works at: http://digitalarchive.gsu.edu/cs_theses

Recommended Citation
Clayton, Arnshea, "The Relative Importance of Input Encoding and Learning Methodology on Protein Secondary Structure Prediction" (2006). *Computer Science Theses.* Paper 19.

This Thesis is brought to you for free and open access by the Department of Computer Science at Digital Archive @ GSU. It has been accepted for inclusion in Computer Science Theses by an authorized administrator of Digital Archive @ GSU. For more information, please contact digitalarchive@gsu.edu.

THE RELATIVE IMPORTANCE OF INPUT ENCODING AND LEARNING METHODOLOGY ON PROTEIN SECONDARY STRUCTURE PREDICTION

by

ARNSHEA CLAYTON

Under the Direction of Yanqing Zhang

ABSTRACT

In this thesis the relative importance of input encoding and learning algorithm on protein secondary structure prediction is explored. A novel input encoding, based on multidimensional scaling applied to a recently published amino acid substitution matrix, is developed and shown to be superior to an arbitrary input encoding. Both decimal valued and binary input encodings are compared. Two neural network learning algorithms, Resilient Propagation and Learning Vector Quantization, which have not previously been applied to the problem of protein secondary structure prediction, are examined. Input encoding is shown to have a greater impact on prediction accuracy than learning methodology with a binary input encoding providing the highest training and test set prediction accuracy.

INDEX WORDS: Neural Networks, Protein Secondary Structure Prediction, Input Encoding, Resilient Propagation, Learning Vector Quantization

# THE RELATIVE IMPORTANCE OF INPUT ENCODING AND LEARNING METHODOLOGY ON PROTEIN SECONDARY STRUCTURE PREDICTION

by

ARNSHEA CLAYTON

A Thesis Submitted in Partial Fulfillment of Requirements for the Degree of

Master of Science

in the College of Arts and Sciences

Georgia State University

2006

Copyright by
Arnshea Clayton
2006

# THE RELATIVE IMPORTANCE OF INPUT ENCODING AND LEARNING METHODOLOGY ON PROTEIN SECONDARY STRUCTURE PREDICTION

by

ARNSHEA CLAYTON

Major Professor: Yanqing Zhang
Committee: Yi Pan
Rajeshekhar Sunderraman

Electronic Version Approved:

Office of Graduate Studies

College of Arts and Sciences

Georgia State University

May 2006

## Acknowledgements

I would like to thank Dr. Yanqing Zhang for his guidance, suggestions and expert advice. I would also like to thank Drs. Pan and Sunderraman for their assistance. Lastly, I would like to thank my wife whose insight into statistical analysis has been a tremendous help.

# Table of Contents

Acknowledgements ..... iv

Table of Contents ..... v

List of Tables ..... vii

List of Figures ..... viii

1 Introduction ..... 1

2 Input Encoding ..... 5

2.1 Amino Acid Encoding ..... 6
2.2 Sammon Mapping ..... 7
3 Learning Algorithms ..... 8

3.1 Resilient Propagation (RPROP) ..... 11
3.2 Learning Vector Quantization (LVQ) ..... 12
3.3 LVQ1 ..... 13
3.4 OLVQ1 ..... 13
3.5 LVQ3 ..... 14
4 Secondary Structure ..... 14

4.1 DSSP ..... 14
4.2 DEFINE ..... 16
4.3 STRIDE ..... 16
5 Architecture ..... 16

5.1 Decision Functions ..... 17
5.2 Single Stage Networks ..... 18
5.3 Three 1-state RPROP networks ..... 18
5.4 3-state RPROP network ..... 19
5.5 3-state LVQ network ..... 19
5.6 Two Stage Networks ..... 20
6 Prediction Accuracy Measures ..... 21

6.1 Cross-Validation ..... 21
6.2 Q3 ..... 22
6.3 Q3* ..... 23
6.4 SOV3 ..... 23

7 Simulation Results ..... 24

7.1 Single stage networks ..... 25
7.2 Three 1-state RPROP networks ..... 25
7.3 One 3-state RPROP network ..... 28
7.4 One 3-state LVQ Network ..... 29
7.5 Two stage networks ..... 31
7.6 Three 1-state RPROP networks into one 3-state RPROP network ..... 32
7.7 One 3-state RPROP network into one 3-state RPROP network ..... 32
7.8 Hybrid LVQ-RPROP network ..... 33
8 Conclusions ..... 35

References ..... 36

Appendix – Amino Acid Encodings ..... 39

Binary ..... 39
Scaled ..... 39
Scaled2 ..... 40
Arbitrary ..... 41

# List of Tables

Table I Summary of training set accuracy. ..... 26

Table II Summary of test set accuracy. ..... 26

Table III Summary of training set accuracy. ..... 27

Table IV Summary of testing set accuracy. ..... 27

Table V Differences in test set accuracy between scaled2 and arbitrary encoding. ..... 28

Table VI Summary of training set accuracy. ..... 29

Table VII Summary of testing set accuracy. ..... 29

Table VIII Summary of test set accuracy for lvq3 algorithm with 384 codebook vectors. ..... 30

Table IX Summary of testing set accuracy. ..... 32

Table X Summary of testing set accuracy. ..... 33

Table XI Summary of training set accuracy. ..... 33

Table XII Summary of test set accuracy. ..... 33

Table XIII Differences in Q3, per sequence Q3 and SOV3 for arbitrary and scaled encodings. ..... 34

# List of Figures

Fig. 1. (a) An alpha helix. (b) A beta sheet. Image taken from [2]. ..... 2

Fig. 2. An example amino acid window. The central amino acid Valine is in bold. This window was taken from a sub-chain of Ubiquitin (1UBQ:C) ..... 6

Fig. 3. (a) real neuron (b) artificial neuron (unit). Image taken from [22]. ..... 9

Fig. 4 Schematic of Prediction Architecture. ..... 17

Fig. 5. Schematic of first stage RPROP network of sub-networks. ..... 18

Fig. 6. 3-state RPROP network. ..... 19

Fig. 7. Schematic of single stage LVQ network. ..... 20

Fig. 8 Per-state test set accuracy as a function of amino acid encoding. ..... 28

Fig. 9. Test set accuracy (Q3) as a function of the number of codebooks, lvq algorithm and amino acid encoding (results for 1536 and 3072 codebooks not shown). ..... 30

Fig. 10. Training set Q3 prediction accuracy as a function of number of codebooks for lvq3 algorithm with scaled encoding. ..... 31

Fig. 11. Per-state prediction accuracy as a function of input encoding. ..... 34

# 1 Introduction

The existence of α-helices and β-sheets, which would later come to be known as protein secondary structure, were predicted in 1951 by Linus Pauling [1] and later observed by X-ray crystallography within the same decade. X-ray crystallography remains the primary experimental method for determining the 3D structure of proteins [2] and is responsible for over 90% of the known protein structures in the largest repository of protein structures, the protein databank (PDB) [3].

Proteins are polymers of amino acids. These polymers, upon post-translational modification, fold into the native form in which they carry out the work of living organisms. This native conformation is described as tertiary (or quaternary in cases where multiple chains combine) structure. Secondary structure describes commonly found structural motifs in tertiary structure. See [2] for a review of secondary structure.

The most common secondary structures are the α helix and β sheet [2]. α helices are characterized by φ and ψ angles of roughly -60 degrees with approximately 3.6 amino acids per complete turn. β strands are characterized by regions of extended backbone conformation with φ=-135 degrees and ψ=135 degrees. Idealized versions of these secondary structures are shown in Fig. 1. There are several automated methods for determining secondary structure including DSSP [1], DEFINE [4] and STRIDE [5]. Of these three DSSP is the most widely used and is the method used by PDB. DSSP defines 7 secondary structure states. Most prediction algorithms reduce these to 3 (Helix, Strand and Coil) states.

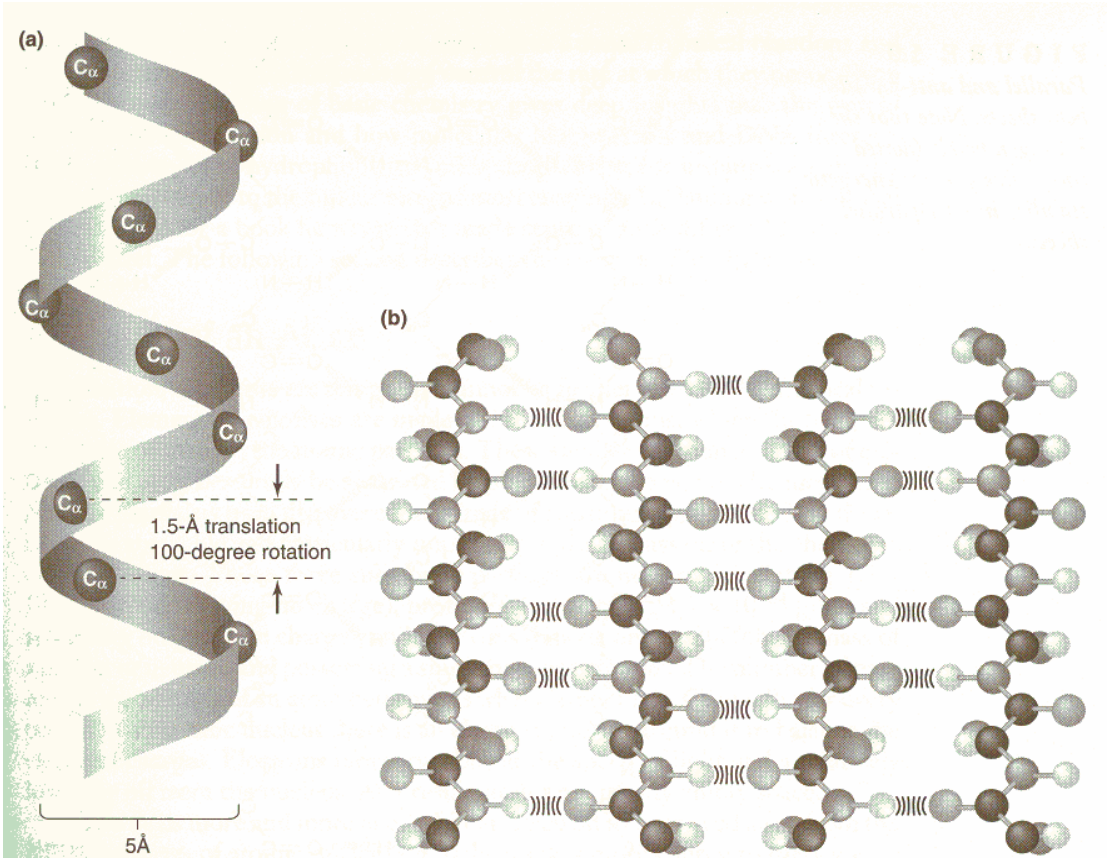


**Fig. 1. (a) An alpha helix. (b) A beta sheet. Image taken from [2].**

Advances in DNA sequencing have resulted in increases in the number of sequenced genomes (and consequently, sequenced protein chains) several orders of magnitude greater than in the preceding decades. The manually intensive and time consuming nature of X-ray crystallography gave rise to the Sequence-Structure gap and motivated the development of automated methods for the prediction of protein secondary structure. Secondary structure prediction methods operate on the amino acid sequence (known as 1D structure) and do not require the 3D structure. Early methods, such as the method of Chou and Fasman [6] and the GOR [7] method were statistical in nature and based largely on probabilities assigned from amino acid proportions in regions of known

secondary structure from small protein sequence databases. Although these methods initially reported accuracy above 70%, when later tested on larger databases the accuracy falls below 60% [8].

The Sequence-Structure gap roughly coincided with the resurgence of interest in neural networks during the 1980s engendered by the discovery of the back-propagation [9] neural network learning algorithm. During the past 15 years neural networks of increasing sophistication and accuracy have been applied to the problem of secondary structure prediction.

In [10] Qian and Sejnowski develop a secondary structure prediction neural network that attains a Q3 accuracy of 64.3%. This result was achieved with two serially cascaded back-propagation networks with binary input encoding and 3 output units. The network was tested on a set of 15 proteins screened for homology with a training set of 91 proteins. The authors manually adjusted weights during training to maximize prediction accuracy. An alternative input encoding was attempted in which physicochemical properties such as charge, size and hydrophobicity were used as input but was not found to improve prediction accuracy. Although the authors explore alternative input encodings the alternatives are limited to distributed input encodings.

In [11] Holley and Karplus achieve 63% Q3 accuracy on a small set of 14 proteins using a single back-propagation network with binary input encoding and 2 output units. The training set contained 48 proteins. The secondary structure prediction for each amino acid is based on an experimentally determined threshold for the output units. This threshold was chosen to maximize predictive accuracy. In the same paper, an

alternative distributed input encoding based on physicochemical properties is reported to achieve a slightly lower accuracy of 61.1%.

In [12] Kneller, et al., report 64% Q3 accuracy using 2 serially cascaded back-propagation networks. A binary input encoding was used. A database of 105 proteins was divided into 91 training set proteins and 14 test set proteins. Spatial units added to the input layer to detect α helix and β strand periodicity increased Q3 accuracy to 65%.

In [13] Chandonia and Karplus achieve 62.64% Q3 accuracy using an iterative combination of structural class and secondary structure back-propagation neural networks on a database of 69 protein chains. The impact of varying the size of the input window and hidden layer is explored with a peak accuracy achieved at a window size of 19 and hidden layer size of 2. A smoothing procedure in which successive output values are averaged resulted in a slight improvement in prediction accuracy.

More recent neural network secondary structure prediction methods make use of evolutionary profile information determined from multiple aligned sequences (see [14-16]). These methods have increased accuracy above 70%. However, there is still benefit from improvements in single sequence input prediction methods particularly for novel amino acids (those without sequence homologues) found in nature and artificially constructed.

This paper explores the relative importance of input encoding and choice of neural network learning algorithm on single sequence neural network secondary structure prediction. A novel local input encoding is developed by applying dimensionality reduction to a Euclidean metric amino acid substitution matrix. Two learning algorithms

which have not previously been applied to the problem of secondary structure prediction are tested: the Resilient Propagation (RPROP) algorithm [17] and the Learning Vector Quantization (LVQ) algorithm [18]. We show that input encoding has a greater effect on prediction accuracy than the choice of neural network learning algorithm. We also show that an input encoding based on evolutionary substitution odds can improve prediction accuracy relative to an arbitrary encoding.

## 2 Input Encoding

Input is presented to Artificial Neural Networks (ANNs) in the form of input patterns. Each input pattern is described by an n-dimensional input vector where n is the number of components used to describe a single pattern. Targets are described by m-dimensional target vectors where m is the number of components used to describe the desired output. Since the neural networks in this paper are predicting secondary structure m will equal the number of secondary structure states.

The protocol for representing a given input or target pattern as a vector is the encoding. In a binary encoding each amino acid is represented by a group of vector components (also called bits). A window of amino acids is represented by a single vector where each position in the window is represented by a single group. The number of components per group corresponds to the number of distinct amino acid types. The component corresponding to the amino acid type at the group position is the only component set to 1 (the rest are 0). Since only a single component per group can be nonzero, this encoding is occasionally called orthogonal encoding.

In a decimal encoding each component of the vector represents a distinct feature (in this case, a separate amino acid) of the input and can take on a range (usually between 0 and 1) of real-numbered values.

The size of the input layer in a neural network is fixed. However, the number of amino acids in a protein sequence can vary from as little as 30 to more than 500 [2]. This necessitates a partitioning of each amino acid sequence into fixed sized windows. In sequence-to-structure networks the amino acid sequence is taken as input and a secondary structure prediction is produced. As shown in the example below, each amino acid in a protein sequence is represented by a fixed sized window that includes the subject amino acid in the center flanked by the surrounding amino acids.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| M | Q | I | F | **V** | K | T | L | T |

**Fig. 2. An example amino acid window. The central amino acid Valine is in bold. This window was taken from a sub-chain of Ubiquitin (1UBQ:C)**

When cascaded networks are used a second stage structure-to-structure network takes the first stage predictions as input, windowed about the subject amino acid, and produces a final secondary structure prediction as output.

The amino acid encodings described in this paper can be found in Appendix A – Amino Acid Encodings.

## 2.1 Amino Acid Encoding

A novel amino acid encoding based on a recently published amino acid similarity matrix (mPAM) [19] is developed. Unlike previous amino acid similarity matrices, mPAM satisfies all three requirements of a distance metric d(x,y):

$$\begin{aligned} &d(x,y) \geq 0, where\, equality\, holds\, only\, if\ x = y \\ &d(x,y) = d(y,x) \\ &d(x,z) \leq d(x,y) + d(y,z) \end{aligned} \tag{1}$$

The LVQ learning algorithm uses a distance metric to classify an input vector into the same class as its nearest codebook vector. The RPROP algorithm does not explicitly depend on distance metrics but may also benefit from the new encoding because this encoding should simplify the search through weight space for weight vectors that induce proper class boundaries on input vectors.

## 2.2 Sammon Mapping

As described in [20] a Sammon Mapping is a multidimensional scaling method often used to provide simplified visual representations of high dimensional data. A Sammon Mapping was used to map the two dimensional amino acid distance matrix mPAM onto one dimension. One dimension is suitable for representation of each amino acid as locations in a single dimension can be described by a single real number.

A Sammon Mapping iteratively minimizes the cost function $E_S$:

$$E_S = \sum_{K \neq l} \frac{\left(d(k,l) - d'(k,l)\right)^2}{d(k,l)} \tag{2}$$

where d(k,l) is the distance between elements k and l in the original vector space and d'(k, l) is the distance between elements k and l in the reduced vectors space. Iteration involves repeatedly, on the order of $10^5$ times the number of samples (in this case, roughly 24 = 20 naturally occurring amino acids + Selenocysteine (U), 2 hybrid amino acids (B = Aspartic Acid or Asparagine and Z = Glutamic acid or Glutamine) and

unknown (X). See [21] for nomenclature), adjusting the one dimensional pair-wise mPAM distance between amino acids k and l $||r_k - r_l||$ where k and l are any two different amino acids. The adjustments are described as follows:

$$\Delta r_k = \lambda \frac{\left(d(k,l) - \|r_k - r_l\|\right)}{\|r_k - r_l\|}\left(r_k - r_l\right) \tag{3}$$

$$\Delta r_l = -\Delta r_k \tag{4}$$

Application of Sammon's Mapping to mPAM resulted in values which were then scaled down to between 0 and 1 to form the scaled (and scaled2) encodings described in Appendix A – Amino Acid Encodings. Values for B and Z in the mPAM matrix were derived by averaging the values for their constituent amino acids. Values for U were set to 7 which equaled the largest distance found between any two other amino acids. Values for X were set to the average over all other distances.

## 3 Learning Algorithms

Neural networks are a biologically inspired class of universal function approximators capable of learning from experience. They are biologically inspired both in terms of structure and manner of processing. However, apart from their utility as models of human cognition, they have proved to be a useful computational paradigm. Structurally, artificial neural networks (ANNs) are comprised of processing units that are analogous to neurons. Each processing unit receives input from other processing units, generates output based on that input (and its own state) then transmits that output to other processing units. This manner of processing is analogous to the functioning of the human

brain wherein a neuron receives electrochemical signals from other neurons across synapses via dendritic tree, internally process the input then transmits electrochemical output through their axon to other neurons (see Fig. 3).

Since the units of neural networks are often grouped into layers neural networks can be broadly classified, in terms of the direction of communication between units, as either recurrent or feed-forward. In feed-forward networks each unit only receives input from units in preceding layers. In recurrent network each unit can receive input from units in any layer. This paper focuses on a further restricted class of feed-forward neural networks in which input is only received from the directly preceding layer.

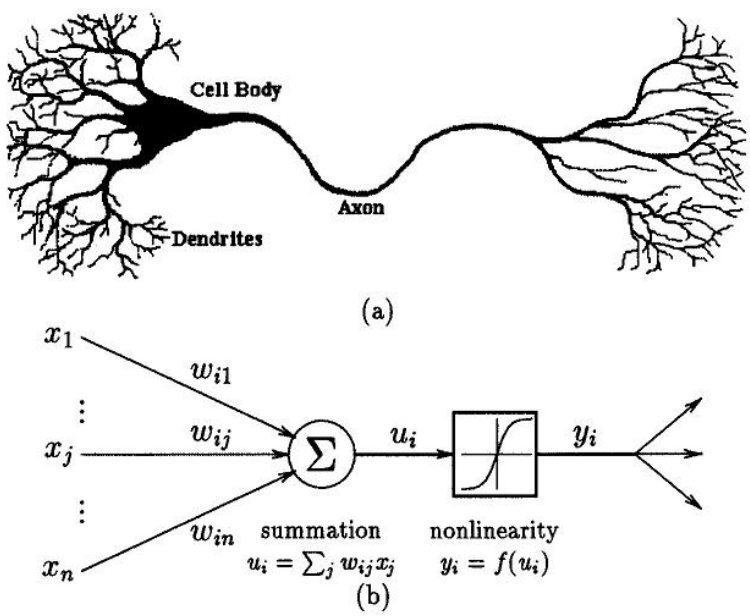


**Fig. 3. (a) real neuron (b) artificial neuron (unit). Image taken from [22].**

It has been shown in [23] that multiple layer feed-forward neural networks (MLFNs) with non-linear activation functions are capable of approximating any function. Although MLFNs are among the slowest to learn [24], their ability to generalize often exceeds that of other methods.

The neural networks discussed in this paper carry out a form of learning known as supervised learning. Supervised learning involves 2 phases; the learning phase (also called the training phase) and the recall phase (also called the prediction phase). During the learning phase a set of examples are presented to the ANN. Each example consists of an input pattern and the desired output pattern (known as the target pattern). The output of the ANN is compared to the target and the weights in the network are adjusted to reduce the difference between the desired output and the actual output (the error). The "supervision" in supervised learning refers to the fact that during training the desired output for each input is known; the network's learning is "supervised" so as to reduce the error. During the recall phase only input patterns are presented to the ANN. This phase is also called the prediction phase because the output produced by the ANN can be considered a prediction since the target output is not presented (and possibly unknown).

The learning algorithm of an ANN is the systematic method by which weights are adjusted to minimize error. One of the most attractive properties of ANNs is their ability to generalize from examples. The goal of generalization is to produce accurate predictions when presented with novel input (input on which it has not been trained). While there are many learning algorithms (see [25]) this paper will focus on Learning Vector Quantization (LVQ) [20] and a modified form of Back-Propagation known as Resilient Propagation (RPROP) [17] .

## 3.1 Resilient Propagation (RPROP)

RPROP [17] is an adaptive learning rate neural network learning algorithm. Adaptive learning rate algorithms automatically vary the rate of weight adjustment (and hence the rate of learning) with the general goal of speeding up convergence. One of the main difficulties encountered with standard back-propagation neural networks is long training times before convergence (or no convergence at all). This is often caused by poor choice of parameters such as the learning rate, momentum, etc… By dropping use of momentum and automatically adjusting the learning rate, RPROP achieves faster convergence while requiring less manual optimization of network parameters.

Whereas standard BP uses fixed proportions (the learning rate) of the error gradient to adjust weights, RPROP introduces a time varying weight step $\Delta_{ij}$ for every weight. When the error gradient changes sign, which indicates the crossing of an extrema, the algorithm reduces the size of the weight step then retracts the most recent step. Like BP, when the error is increasing (indicated by a positive error gradient) the weights are reduced and when the error is decreasing (indicated by a negative error gradient) the weight are increased. However, unlike BP, the size of the adjustment is no longer a fixed percentage of the error gradient. The size of the weight step $\Delta_{ij}$ is adjusted as follows:

$$\Delta_{ij}^{(t)} = \begin{cases} \eta^{+} * \Delta_{ij}^{(t-1)}, \text{if} \left(\frac{\partial \mathrm{E}}{\partial \mathrm{w}_{\mathrm{ij}}}\right)^{(t-1)} * \left(\frac{\partial E}{\partial w_{ij}}\right)^{(t)} > 0 \\ \eta^{-} * \Delta_{ij}^{(t-1)}, \text{if} \left(\frac{\partial \mathrm{E}}{\partial \mathrm{w}_{\mathrm{ij}}}\right)^{(t-1)} * \left(\frac{\partial E}{\partial w_{ij}}\right)^{(t)} < 0 \\ \Delta_{ij}^{(t-1)}, \text{else} \end{cases} \tag{5}$$

where $0 < \eta^{-} < 1 < \eta^{+}$, $\Delta_{ij}^{(t)}$ = weight step for $w_{ij}$ at time t and $\Delta_{ij}^{(t-1)}$ = previous weight step. The weights themselves are

## 3.2 Learning Vector Quantization (LVQ)

Learning Vector Quantization [20] is a neural network method based on classical vector quantization and subspace classification. In classical vector quantization, a continuous signal is approximated by the closest codebook vectors where each codebook vector is the discrete representation of a point in continuous signal space. In subspace classification, n-dimensional data is represented by the nearest lower dimensional orthogonal projection.

Under LVQ, a set of codebook vectors (also called models) $m_i$ are each associated with one of the desired classes $S_i$. Each class contains more than one codebook vector. When an input vector x is presented a winner $m_c$ is chosen on the basis of a distance metric (the closest $m_i$ wins). Typically Euclidean distance is used. While there are several variations of this method, some of which are discussed below, the general idea is to adjust the winning codebook vector to more closely approximate its class $S_i$. That is, if

both $m_c$ and x are in the same class then $m_c$ is moved closer to x otherwise it is moved away.

In standard neural network parlance each hidden unit represents a codebook whose incoming weights are the components of the codebook vector. Each output unit represents a separate class. The output units only receive input from hidden units in the same class.

## 3.3 LVQ1

In the first LVQ method, LVQ1[18], the codebook vectors $m_i$ are updated as follows:

$$m_i(t+1) = \begin{cases} m_i(t) + \alpha(t)\big(x(t) - m_i(t)\big) \\ \quad \text{, if i = c and x(t) and } m_c \text{ belong to the same class} \\ m_i(t) - \alpha(t)\big(x(t) - m_i(t)\big) \\ \quad \text{, if i = c and x(t) and } m_c \text{ belong to different classes} \\ m_i(t) \\ \quad \text{, if i} \neq \text{c} \end{cases} \tag{6}$$

where $0 < \alpha(t) < 1$ and $\alpha(t)$ (the learning rate) decreases monotonically over time and c is the index of the winning codebook vector.

## 3.4 OLVQ1

The first optimized learning vector quantization (OLVQ1) [20] algorithm optimally adjusts the learning rate $\alpha(t)$. Optimality is achieved when prediction accuracy cannot be improved by changing $\alpha(t)$. This can be recursively determined as follows:

$$\alpha_c(t) = \frac{\alpha_c(t-1)}{1 + s(t)\alpha_c(t-1)} \tag{7}$$

where s(t) = +1 when the current input pattern x is in the same class as $m_c$ and s(t) = -1 when x is not in the same class as $m_c$. The model updates are the same as those defined in section 3.3 except that α is replaced with $\alpha_c$ which is recomputed with each pattern presentation.

## 3.5 LVQ3

The third learning vector quantization (LVQ3) [20] algorithm adjusts both the nearest codebook vector and the next nearest codebook vector as follows:

$$\begin{aligned} m_i(t) &= m_i(t) - \alpha(t)\left[x(t) - m_i(t)\right] \\ m_j(t) &= m_j(t) + \alpha(t)\left[x(t) - m_i(t)\right] \end{aligned} \tag{8}$$

where $m_i$ and $m_j$ are the closest codebook vectors to x where x is in the same class as $m_j$ and in a different class than $m_i$. x is subject to the additional restriction:

$$m_k(t+1) = m_k(t) + \varepsilon\alpha(t)\left[x(t) - m_k(t)\right] \tag{9}$$

for k in {i, j} if x, $m_i$ and $m_j$ belong to the same class.

# 4 Secondary Structure

## 4.1 DSSP

Dictionary of Protein Secondary Structure (DSSP) [1] is the most widely used algorithm [5] for the determination of secondary structure. It is a hierarchical method whose root parameter is the presence or absence of a hydrogen bond. This parameter was chosen for

simplicity; the presence or absence of a hydrogen bond, once defined, is binary whereas use of backbone torsion angles φ and ψ introduces four additional parameters (the four angles of a rectangle in the φ and ψ plane) for each type of secondary structure.

On the basis of an application specific definition of a hydrogen bond (described in [1]), elementary secondary structural elements are defined as follows:

$$n - turn(i) = Hbond(i, i+n), n = 3,4,5 \qquad \text{Eq 4.1}$$

$$\text{Parallel Bridge}(i, j) = \begin{bmatrix} \text{Hbond}(i-1, j) \text{ and Hbond}(j, i+1) \\ \text{or} \\ \text{Hbond}(j-1, i) \text{ and Hbond}(i, j+1) \end{bmatrix}$$

$$\text{Antiparallel Bridge}(i, j) = \begin{Bmatrix} \text{Hbond}(i, j) \text{ and Hbond}(j, i) \\ \text{or} \\ \text{Hbond}(i-1, j+1) \text{ and Hbond}(j-1, i+1) \end{Bmatrix} \qquad \text{Eqs 4.2}$$

where hbond(i,j) indicates a hydrogen bond between the carboxyl group of the ith residue and the amino group of the jth residues.

The next level in the hierarchical definition of secondary structure is the cooperative structure level. Cooperative structural elements are defined in terms of elementary structural elements. Ambiguity is resolved by priority assignment. Cooperative secondary structural elements are defined as follows:

$$\begin{aligned} 4\text{-helix}(i,i+3) &= \{4\text{-turn}(i-1) \text{ and } 4\text{-turn}(i)\} \\ 3\text{-helix}(i,i+2) &= \{3\text{-turn}(i-1) \text{ and } 3\text{-turn}(i)\} \\ 5\text{-helix}(i,i+5) &= \{5\text{-turn}(i-1) \text{ and } 5\text{-turn}(i)\} \end{aligned} \qquad \text{Eqs 4.3}$$

$$\begin{aligned} \text{ladder} &= \text{set of one or more consecutive bridges of identical type} \\ \text{sheet} &= \text{set of one or more ladders connected by shared residues} \end{aligned} \qquad \text{Eqs 4.4}$$

where 4-helix (α helix), 3-helix ($3_{10}$ helix) and 5-helix (π helix) are identified by the codes H, G and I respectively. Single bridges are labeled B. All other ladder residues are labeled E.

## 4.2 DEFINE

DEFINE [4] is an automatic method for the determination of secondary structure based on idealized secondary structure $C^{\alpha}$ distance matrices. The $C^{\alpha}$ distance matrix for the target sequence is computed and residues are assigned to the secondary structure class with the most similar ideal distance matrix.

## 4.3 STRIDE

STRIDE [5] is an algorithm for assigning secondary structure on the basis of both hydrogen bond energy and backbone torsion angles. The x-ray crystallography structures present in the protein data bank (PDB) [3] were analyzed to produce a range of φ and ψ torsion angles as well as hydrogen bond energies most representative of existing secondary structural classifications.

# 5 Architecture

There are three major components of the prediction architecture.

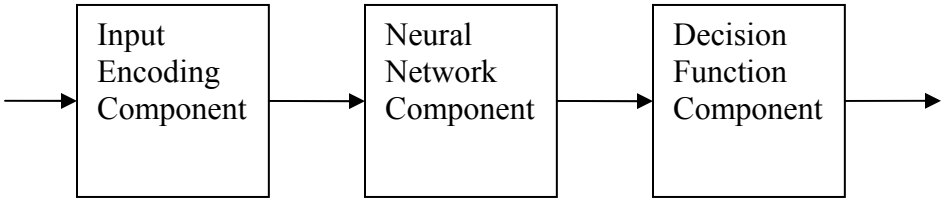


**Fig. 4 Schematic of Prediction Architecture.**

The Input Encoding component partitions protein sequences and their corresponding secondary structure states into fixed sized windows and converts these windows into input/target patterns suitable for training (and testing) in the neural network component.

The Neural Network component predicts the secondary structure of the central amino acid in the input window. This may be done directly by a single sequence-to-structure network or in stages with sequence-to-structure and structure-to-structure networks cascaded serially.

The Decision Function component is responsible for turning the output of the neural network component into secondary structure state predictions. There are 3 secondary structure states which are represented as H, E and C. Neural network output however, is numerical. The decision function component applies several decision functions to convert the numerical output of the neural network component into one of the secondary structure states.

## 5.1 Decision Functions

Two decision functions were employed in this project. The first, df_maxout, assigns the prediction to the state corresponding to the output unit with the largest value. The

second, df_hethresh, assigns the prediction by comparing the output of the H and E units to a variable threshold. This decision function extends the use of the output threshold, first introduced in [13], by incorporating a separate threshold for both the H and E states. The prediction accuracy is determined as both the H and E thresholds vary from .2 to .5 . The thresholds that minimize the standard deviation of per-state accuracy (also known as qindex) are chosen. If neither the H nor E output units exceed their respective thresholds then C is assigned. Otherwise, the larger of the 2 states (H or E) is assigned.

## 5.2 Single Stage Networks

Three different single stage sequence-to-structure networks were compared: three 1-state RPROP networks, one 3-state RPROP network and one 3-state LVQ network.

## 5.3 Three 1-state RPROP networks

Depicted in Fig. 5, each RPROP sub-network is trained to recognize a single secondary structure class and has a single output. The network with the greatest output determines the secondary structure prediction.

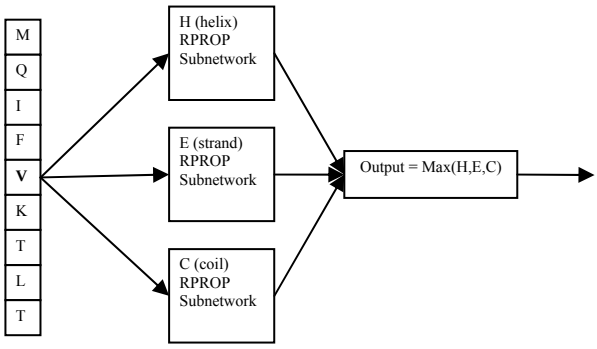


**Fig. 5. Schematic of first stage RPROP network of sub-networks.**

## 5.4 3-state RPROP network

The 3-state RPROP network, depicted below, learns to distinguish all three secondary structure states (H, E and C). The output unit, $y_i$, with the greatest value determines the secondary structure prediction of the central amino acid $x_i$.

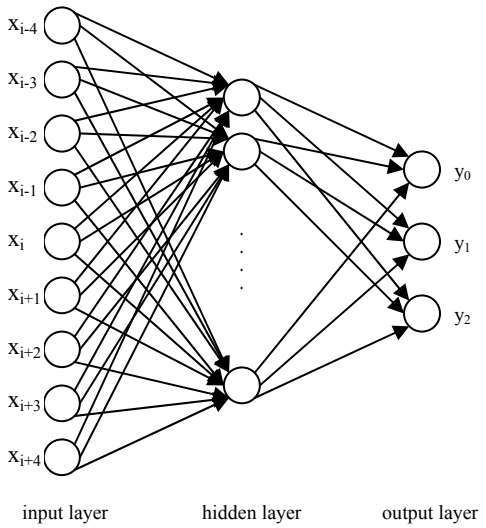


**Fig. 6. 3-state RPROP network.**

## 5.5 3-state LVQ network

The LVQ network is a single stage network with 3 outputs. The classification of an amino acid is based on the class of the nearest codebook vector $m_c$ after training. Since LVQ networks associate input with a class (in this case, each class is a secondary structure state) there is no numerical output available for the decision function component. The network prediction is used directly.

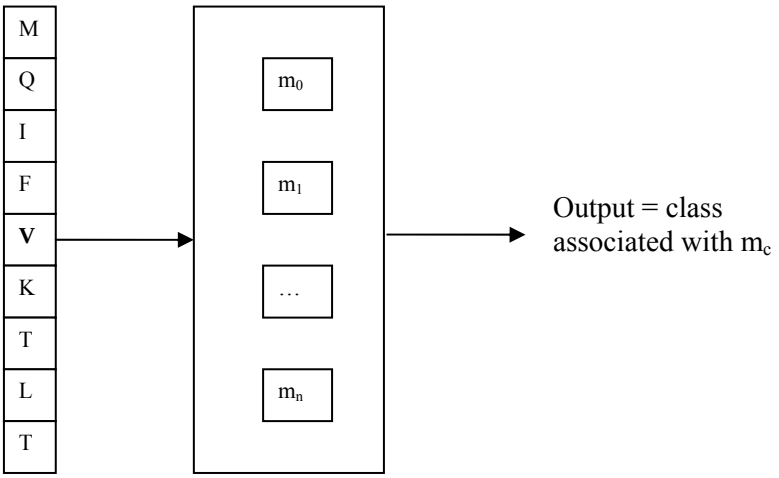


**Fig. 7. Schematic of single stage LVQ network.**

**Codebook vectors mi shown in center. The class of the codebook mc closest to the input vector is the predicted class.**

## 5.6 Two Stage Networks

Two stage networks are comprised of a sequence-to-structure stage and a structure-to-structure stage. The output of the sequence-to-structure stage network is presented as input into the second stage structure-to-structure network. The addition of a structure-to-structure network is intended to provide a network that learns the association between adjacent amino acids as the sequence-to-structure network is not necessarily trained on the amino acids in sequence order. It is presumed that the first stage networks cannot learn this association if they are trained on randomly selected amino acids which are not necessarily adjacent. Earlier work ([10], [16], [26]) has shown slight improvement in prediction accuracy with the addition of a structure-to-structure network.

Three such configurations were examined: three 1-state RPROP networks cascaded into a single 3-state RPROP network, a 3-state RPROP network cascaded into another 3-

state RPROP network and a hybrid of an LVQ network + three 1-state RPROP networks cascaded into a 3-state RPROP network.

## 6 Prediction Accuracy Measures

A variety of measures have been employed to describe the accuracy of secondary structure prediction methods. While early neural network secondary structure prediction methods included a Matthews correlation coefficient, this measure is no longer widely used.

The Comprehensive Assessment of Secondary Structure Prediction (CASP) is a multi-year project to provide an objective comparison of protein structure prediction methods. 3-D structures are submitted to CASP before they are made public. Prediction methods are then tested against these pre-release structures. Since the structures have not yet been made public, this procedure reduces the possibility of inflated prediction accuracy due to training bias and provides a blind evaluation of all submitted methods. Four of the first five CASPs included a secondary structure prediction category (see [27-30]). The current CASP (CASP VI) has dropped the secondary structure category. The prediction accuracy measures used in this paper were taken from the metrics used in the CASPs that included a secondary structure category.

### 6.1 Cross-Validation

Given a set of protein sequences properly screened to remove any sequence homologues the ideal method of validating the prediction accuracy is the leave-one-out (jackknife) method. Under the leave-one-out method tunable parameters (in the case of neural

networks the weights are the tunable parameters) are trained on N-1 of N proteins then tested against the remaining protein. This is repeated N times so that each protein is tested exactly once. The computational overhead of training neural networks makes leave-one-out implausible for all but the smallest datasets. Cross-validation [31] is a widely used compromise method that balances the desire to avoid inflated accuracy due to training set bias with the computational intensity of full-blown leave-one-out validation. Under cross-validation the network is trained on (M-1)(N/M) proteins then tested on the remaining (N/M) proteins where M is the fold of validation. This is repeated each time with a different set of (N/M) proteins left out of training for testing purposes until every protein has been tested at least once.

## 6.2 Q3

Q3 measures the percentage of correctly identified residues for all secondary structure classes (H, E and C). This method is the simplest measure of prediction accuracy. However, because it places undue emphasis on residue accuracy while secondary structure is defined in terms of segments, it can provide an inaccurate estimate of the relative accuracy of prediction methods. A separate Qindex for each state (H, E and C) is also provided. Q3 measures are defined as follows:

$$
\begin{aligned}
Qh &= \frac{p_h}{n_h} \\
Qe &= \frac{p_e}{n_e} \\
Qc &= \frac{p_c}{n_c} \\
Q3 &= \frac{p_h + p_e + p_c}{N}
\end{aligned}
\tag{5}
$$

where $Q_{\{h,e,c\}}$ is the per-state accuracy, $p_{\{h,e,c\}}$ is the number of helices, strands or coils correctly predicted, $n_{\{h,e,c\}}$ is the total number of helices, strands or coils and N is the total number of residues.

## 6.3 Q3*

Q3* is the average of Q3s calculated for each sequence separately. Since the sequences have different lengths this is an un-weighted average.

## 6.4 SOV3

Segment Overlap, originally defined in [32] and updated in [33], provides an indication of how well secondary structure segments were predicted. In some protein sequences, a Q3 accuracy greater than 50% can be achieved by simply predicting every residue to be a helix. SOV3 on the other hand would result in a much lower score. SOV3 is defined as follows:

$$
\begin{aligned}
&S(i) = \{(s_1, s_2) : s_1 \cap s_2 \neq \varnothing, \mathrm{s_1 and\ s_2\ both\ in\ conformational\ state\ i}\} \\
&S'(i) = \{s_1 : \forall s_2, s_1 \cap s_2 = \varnothing, \mathrm{s_1 and\ s_2\ both\ in\ conformational\ state\ i}\} \\
&N(i) = \left(\sum_{S(i)} len(s_1)\right) + \left(\sum_{S'(i)} len(s_1)\right) \\
&N = \sum_{i \in \{H,E,C\}} N(i) \\
&\delta(s_1, s_2) = \min \left\{ \begin{array}{l} (\max ov(s_1, s_2) - \min ov(s_1, s_2)), \\ \min ov(s_1, s_2), \\ \mathrm{int}\left( len(s_1) / 2 \right), \\ \mathrm{int}\left( len(s_2) / 2 \right) \end{array} \right\} \\
&Sov = 100 * \left\{ \frac{1}{N} \sum_{i \in \{H,E,C\}} \sum_{S(i)} \left( \frac{\min ov(s_1, s_2) + \delta(s_1, s_2)}{\max ov(s_1, s_2)} \times len(s_1) \right) \right\}
\end{aligned}
\tag{6}
$$

where ($s_1$,$s_2$) denote an overlapping pair of segments, S(i) = the set of all overlapping pairs of segments ($s_1$,$s_2$) in state I and S'(i) = the set of all segments $s_1$ for which there is no overlapping segment $s_2$ in state i.

## 7 Simulation Results

The networks were trained on the Rost and Sander dataset (rs126) (described in [16, 32, 34, 35] as dataset 2b). This set of 126 protein chains were carefully selected to minimize internal homology. While there exist other datasets more stringently screened for internal homology there are a plethora of differing screening methodologies with none currently accepted as standard. RS126 was chosen because it is widely used which allows for better comparisons with earlier work.

3-fold cross validation was performed by partitioning rs126 into 3 subsets (2 sets with 40 sequences, 1 set with 46). Each network configuration was trained on 2 of the 3

subsets then tested on the remaining subset. This was repeated with 2 different subsets until each partition was tested at least once. The prediction accuracies are the averages obtained during cross-validation.

The experiments were conducted on an Intel Pentium IV 3.0GHZ processor. The time to execute 100 epochs of training varied from 2 to 4 minutes wall clock time depending on the input encoding and hidden layer size. The Input Encoding component was written in PERL. The Matlab 6.5R13 Neural Network Toolbox was used to carry out all RPROP simulations. All RPROP simulations were executed using 'early stopping' to maximize predictive accuracy. The LVQ networks were executed using version 3.1 of the Helsinki University of Technology lvq_pak software package. All decision functions were written in C++.

## 7.1 Single stage networks

The single stage networks produced secondary structure predictions based directly on the amino acid sequence. The RPROP networks had an initial weight step size of .07, weight increment of 1.2, weight decrement of .5 and maximum weight change of 50.

## 7.2 Three 1-state RPROP networks

Input window sizes of 9, 13 and 17 were used. Hidden layer sizes of 0 (no hidden layer), 2 and 5 were used. Networks were trained for a maximum of 100 epochs though none reached this threshold because of 'early stopping'.

**Table I Summary of training set accuracy.**

**Only the best results for each encoding (based on df_maxout) are shown. See appendix for full results.**

| encoding | winsize | hidden layer size | Q3 | SOV3 | qH | qE | qC |
|---|---|---|---|---|---|---|---|
| arbitrary | 17 | 5 | 0.511 | 0.176 | 0.063 | 0.036 | 0.964 |
| binary | 17 | 5 | 0.675 | 0.537 | 0.603 | 0.428 | 0.822 |
| scaled | 17 | 2 | 0.506 | 0.151 | 0.019 | 0.027 | 0.982 |
| scaled2 | 13 | 5 | 0.524 | 0.282 | 0.034 | 0.190 | 0.941 |

**Table II Summary of test set accuracy.**

**Only the best results for each encoding (based on df_maxout) are shown. See appendix for full results. Peak Q3 prediction accuracy is shown in bold.**

| encoding | winsize | hidden layer size | Q3 | SOV3 | qH | qE | qC |
|---|---|---|---|---|---|---|---|
| arbitrary | 17 | 5 | 0.510 | 0.168 | 0.058 | 0.034 | 0.963 |
| **binary** | **17** | **0** | **0.633** | **0.514** | **0.565** | **0.372** | **0.785** |
| scaled | 9 | 5 | 0.510 | 0.183 | 0.023 | 0.054 | 0.973 |
| scaled2 | 13 | 5 | 0.526 | 0.287 | 0.030 | 0.194 | 0.941 |

The first stage RPROP network achieved a peak Q3 prediction accuracy of 63.3% with the binary encoding. The binary encoding achieves a 12.3% improvement over the scaled encoding. Somewhat unexpectedly the arbitrary encoding Q3 training set accuracy (51.1%) was higher than the scaled encoding (50.6%). As pointed out in [11] and [10] high training set accuracy not reflected in higher testing set accuracy can suggest memorization of the training set patterns instead of generalization of common features. This memorization may explain the reduced testing set prediction accuracy relative to the scaled2 encoding.

It is clear from the imbalance in qH, qE and qC that the arbitrary, scaled and scaled2 encodings derive much of their predictive accuracy from coil prediction. The binary encoding however has a much more balanced per state predictive accuracy. To address this problem an alternative decision function, df_hethresh, was devised.

**Table III Summary of training set accuracy.**

**Only the best results for each encoding (based on df_hethresh) are shown. See appendix for full results.**

| encoding | winsize | hidden layer size | Q3 | SOV3 | qH | qE | qC |
|---|---|---|---|---|---|---|---|
| arbitrary | 13 | 5 | 0.431 | 0.288 | 0.433 | 0.422 | 0.434 |
| binary | 17 | 5 | 0.640 | 0.517 | 0.640 | 0.643 | 0.639 |
| scaled | 17 | 5 | 0.412 | 0.266 | 0.301 | 0.447 | 0.454 |
| scaled2 | 9 | 5 | 0.435 | 0.288 | 0.389 | 0.451 | 0.451 |

**Table IV Summary of testing set accuracy.**

**Only the best results for each encoding (based on df_hethresh) are shown. See appendix for full results. Best results in bold.**

| encoding | winsize | hidden layer size | Q3 | SOV3 | qH | qE | qC |
|---|---|---|---|---|---|---|---|
| arbitrary | 13 | 5 | 0.431 | 0.279 | 0.426 | 0.433 | 0.433 |
| **binary** | **13** | **0** | **0.597** | **0.520** | **0.600** | **0.600** | **0.594** |
| scaled | 9 | 5 | 0.404 | 0.226 | 0.366 | 0.395 | 0.430 |
| scaled2 | 9 | 5 | 0.433 | 0.285 | 0.363 | 0.465 | 0.459 |

The binary encoding produces the most accurate prediction under both decision functions. The decision function df_hethresh minimizes the spread in per-state predictive accuracy. Although this reduces Q3 it increases SOV3 and, by increasing qH and qE, substantially increases the reliability of Helix and Strand predictions.

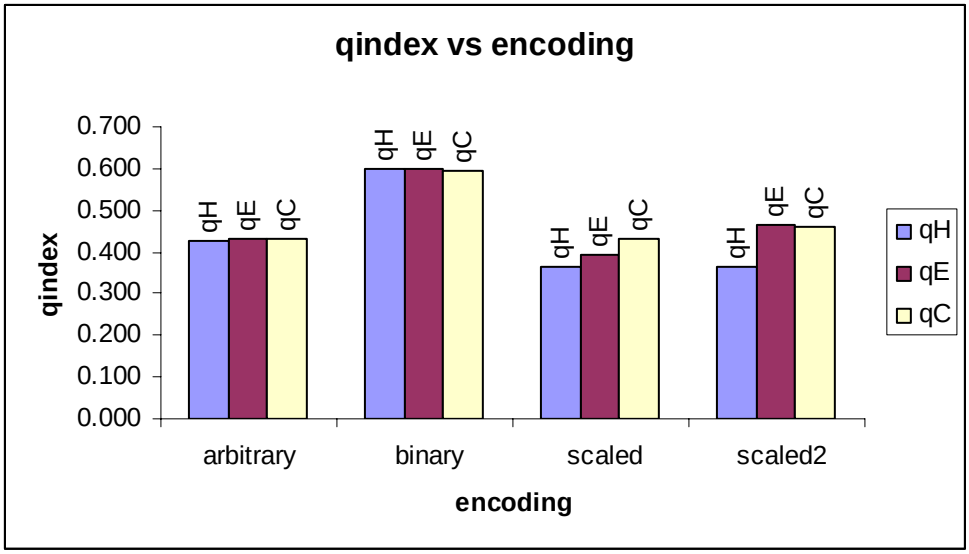


**Fig. 8 Per-state test set accuracy as a function of amino acid encoding.**

The scaled2 encoding has a higher sheet and coil accuracy than the arbitrary encoding. However, the binary encoding Qindices exceed all others. Qindex calculated based on df_hethresh decision function. Only the Qindices of network configuration with highest q3 are shown.

**Table V Differences in test set accuracy between scaled2 and arbitrary encoding.**

| Encoding | Q3 | SOV3 |
|---|---|---|
| Arbitrary | .0.510 | 0.168 |
| Scaled | 0.526 | 0.287 |
| Δ | 0.016 | 0.119 |

## 7.3 One 3-state RPROP network

Input window sizes of 9, 13 and 17 were used. Hidden layer sizes of 0 (no hidden layer), 2 and 5 were used. Networks were trained for a maximum of 100 epochs though none reached this threshold because of ‘early stopping’.

**Table VI Summary of training set accuracy.**

**Only the best results for each encoding (based on df_maxout) are shown. See appendix for full results.**

| encoding | winsize | hidden layer size | Q3 | SOV3 | qH | qE | qC |
|---|---|---|---|---|---|---|---|
| arbitrary | 17 | 5 | 0.506 | 0.127 | 0.020 | 0.018 | 0.985 |
| binary | 17 | 5 | 0.679 | 0.531 | 0.608 | 0.452 | 0.815 |
| scaled | 9 | 5 | 0.508 | 0.198 | 0.005 | 0.074 | 0.973 |
| scaled2 | 13 | 0 | 0.519 | 0.261 | 0.024 | 0.186 | 0.936 |

**Table VII Summary of testing set accuracy.**

**Only the best results for each encoding (based on df_maxout) are shown. See appendix for full results. Best results in bold.**

| encoding | winsize | hidden layer size | Q3 | SOV3 | qH | qE | qC |
|---|---|---|---|---|---|---|---|
| arbitrary | 13 | 5 | 0.508 | 0.155 | 0.028 | 0.032 | 0.974 |
| **binary** | **17** | **0** | **0.632** | **0.518** | **0.563** | **0.384** | **0.779** |
| scaled | 9 | 5 | 0.508 | 0.201 | 0.004 | 0.078 | 0.971 |
| scaled2 | 13 | 0 | 0.519 | 0.254 | 0.021 | 0.188 | 0.935 |

As is the case for the single state networks, the binary encoding produces the most accurate prediction.

## 7.4 One 3-state LVQ Network

The parameters of the first stage LVQ network used in the final network ensemble were selected for highest predictive accuracy. The OLVQ1 and LVQ3 algorithms were compared. The number of codebook vectors that were compared: 92, 192, 384, 768, 1536 and 3072. The OLVQ1 algorithm was executed for 10,000 cycles. The LVQ3 algorithm was executed for 40,000 cycles.

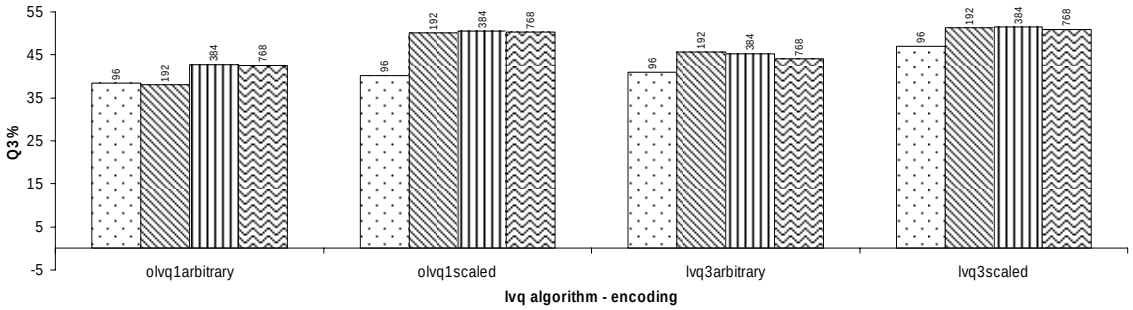


**Fig. 9. Test set accuracy (Q3) as a function of the number of codebooks, lvq algorithm and amino acid encoding (results for 1536 and 3072 codebooks not shown).**

As can be seen in Fig. 9 the LVQ3 algorithm with a codebook size of 384 produced the highest prediction accuracy (51.5%).

**Table VIII Summary of test set accuracy for lvq3 algorithm with 384 codebook vectors.**

| Winsize | Encoding | Q3 |
|---|---|---|
| 9 | Arbitrary | .452 |
| 9 | Scaled | .515 |

Training set accuracy continued to increase with increasing number of codebook vectors as seen in Fig. 10. However training set accuracy increases obtained with more than 384 codebook vectors resulted in decreased prediction accuracy.

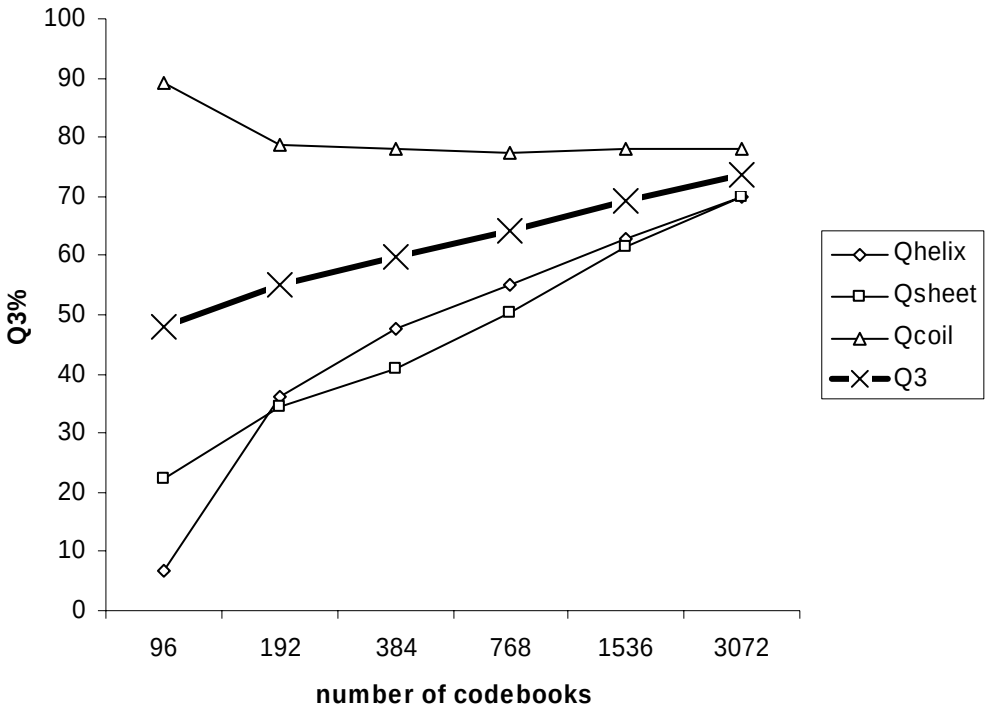


**Fig. 10. Training set Q3 prediction accuracy as a function of number of codebooks for lvq3 algorithm with scaled encoding.**

## 7.5 Two stage networks

Two stage networks produce secondary structure predictions based on the predictions of the first stage networks. In the case of the Hybrid LVQ-RPROP network, only the final secondary structure prediction of the sequence-to-structure networks is used. In the remaining cases all first stage output values are presented to the second stage structure-to-structure network. Except where noted the RPROP network parameters for the two stage networks are the same as for the single stage networks.

## 7.6 Three 1-state RPROP networks into one 3-state RPROP network

The first stage of this configuration is described in section 7.2. The following parameters describe the second stage structure-to-structure network. Input window sizes of 9, 13 and 17 were used. Hidden layer sizes of 0 (no hidden layer), 2 and 5 were used. Networks were trained for a maximum of 100 epochs though none reached this threshold because of 'early stopping'.

**Table IX Summary of testing set accuracy.**

**Only the best results for each encoding (based on df_maxout) are shown. See appendix for full results. Best results in bold.**

| encoding | winsize | hidden layer size | Q3 | SOV3 | qH | qE | qC |
|---|---|---|---|---|---|---|---|
| arbitrary | 17 | 5 | 0.505 | 0.116 | 0.025 | 0.015 | 0.979 |
| **binary** | **17** | **5** | **0.630** | **0.504** | **0.567** | **0.355** | **0.784** |
| scaled | 9 | 5 | 0.508 | 0.183 | 0.000 | 0.094 | 0.964 |
| scaled2 | 13 | 5 | 0.521 | 0.240 | 0.000 | 0.179 | 0.955 |

The addition of a structure-to-structure network does not appear to increase prediction accuracy for this network configuration.

## 7.7 One 3-state RPROP network into one 3-state RPROP network

The first stage of this configuration is described in section 7.3. The following parameters describe the second stage structure-to-structure network. Input window sizes of 9, 13 and 17 were used. Hidden layer sizes of 0 (no hidden layer), 2 and 5 were used. Networks were trained for a maximum of 100 epochs though none reached this threshold because of 'early stopping'.

**Table X Summary of testing set accuracy.**

**Only the best results for each encoding (based on df_maxout) are shown. See appendix for full results. Best results in bold.**

| encoding | winsize | hidden layer size | Q3 | SOV3 | qH | qE | qC |
|---|---|---|---|---|---|---|---|
| arbitrary | 13 | 5 | 0.510 | 0.137 | 0.060 | 0.013 | 0.968 |
| **binary** | **13** | **5** | **0.632** | **0.495** | **0.574** | **0.380** | **0.776** |
| scaled | 17 | 5 | 0.503 | 0.199 | 0.037 | 0.029 | 0.964 |
| scaled2 | 17 | 5 | 0.508 | 0.210 | 0.042 | 0.084 | 0.946 |

The addition of a structure-to-structure network does not improve overall Q3 but appears to slightly improve the accuracy of helix prediction (0.563 vs. 0.574).

## 7.8 Hybrid LVQ-RPROP network

The second stage of the Hybrid RPROP network had 18 or 26 input units (depending on the window size of the corresponding first stage networks) and 32 hidden units. This network was trained with stage 1 training set predictions then tested with stage 1 testing set predictions. The input was the final state prediction (either H, E or C) from both first stage networks.

**Table XI Summary of training set accuracy.**

**Encoding describes amino acid encoding used in corresponding first stage networks.**

| winsize | encoding | Q3 | Q3stddev | Q3* | Q3stddev* | SOV3 | SOV3stddev |
|---|---|---|---|---|---|---|---|
| 9 | arbitrary | .610 | .022 | .622 | .023 | .547 | .039 |
| 9 | scaled | .637 | .012 | .652 | .012 | .600 | .015 |

**Table XII Summary of test set accuracy.**

**Peak prediction accuracy (Q3 of 52.2%) is shown in bold.**

| winsize | encoding | Q3 | Q3stddev | Q3* | Q3stddev* | SOV3 | SOV3stddev |
|---|---|---|---|---|---|---|---|
| 9 | arbitrary | .501 | .010 | .509 | .016 | .439 | .027 |
| **9** | **scaled** | **.522** | **.010** | **.539** | **.007** | **.485** | **.010** |

The second stage RPROP network achieved a peak prediction accuracy of 52.2% with a window size 9 and scaled encoding.

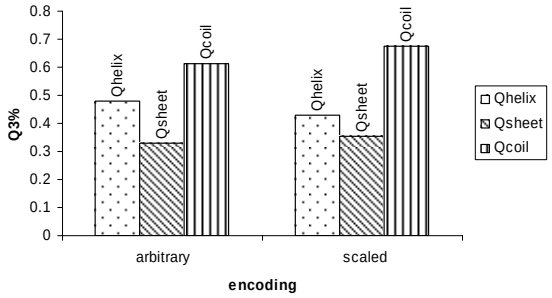


**Fig. 11. Per-state prediction accuracy as a function of input encoding.**

**Table XIII Differences in Q3, per sequence Q3 and SOV3 for arbitrary and scaled encodings.**

| | Q3 | Q3* | SOV3 |
|---|---|---|---|
| Arbitrary | .501 | .509 | .439 |
| Scaled | .522 | .539 | .485 |
| Δ | .021 | .03 | .046 |

# 8 Conclusions

When using a decimal encoding for amino acids a lower dimensional projection of mPAM, an amino acid substitution matrix that satisfies the requirements of a Euclidean distance metric, improves prediction accuracy. The use of scaled mPAM encoding does not appear to improve prediction accuracy for RPROP networks. Improvement might be observed with larger hidden layer sizes and/or discontinuing use of early stopping.

The impact of the scaled mPAM encoding on prediction accuracy is substantial for the LVQ networks. The scaled encoding increases prediction accuracy by 6.3% (51.5% vs. 45.2%) for a window size of 9 and 5.1% (50.6% vs. 45.5%) for a window size of 13 in an LVQ3 network with 384 codebook vectors. This is likely because the LVQ algorithms explicitly use Euclidean distance to determine the closest codebook vector and the scaled mPAM encoding of input vectors reflects the evolutionary similarity expressed in the mPAM amino acid substitution matrix.

Unfortunately adding a second stage structure-to-structure network does not appear to improve overall accuracy for networks with decimal input encoding. However, a binary input encoded network does benefit from an additional stage for helix prediction accuracy (~1-2% improvement). The former finding is inconsistent with previous work ([10] and [12]) in which 1% improvements were obtained by the addition of a second stage network. The lack of an improvement observed in the hybrid LVQ-RPROP network may be due to the manner in which the first stage predictions are presented to the second stage. In the earlier work cited above the second stage network received all three

output values for each amino acid position. In this paper the second stage network received only the final secondary structure prediction for each amino acid position. This arrangement provides no information about the relative strength of each prediction making it difficult for the second stage network to learn the relationship between first stage prediction strength and prediction accuracy.

Although prediction accuracy was improved by the amino acid encoding described in this paper, the highest prediction accuracy was achieved using a binary encoding. This suggests that additional understanding of the significance of input encoding as it relates to secondary structure prediction could be gleaned from an analysis of the clustering properties of known protein sequences when represented by different encoding schemes. Such an analysis could guide the development of alternative input encodings, perhaps based on different substitution matrices or generated with different scaling methods. Greater improvements in prediction accuracy might be obtained with alternative input encodings.

## References

[1] W. Kabsch and C. Sander, "Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features," *Biopolymers*, vol. 22, pp. 2577-637, 1983.

[2] D. E. Krane and M. L. Raymer, *Fundamental concepts of bioinformatics*. San Francisco: Benjamin Cummings, 2003.

[3] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne, "The Protein Data Bank," *Nucleic Acids Res*, vol. 28, pp. 235-42, 2000.

[4] F. M. Richards and C. E. Kundrot, "Identification of structural motifs from protein coordinate data: secondary structure and first-level supersecondary structure," *Proteins*, vol. 3, pp. 71-84, 1988.

[5] D. Frishman and P. Argos, "Knowledge-based protein secondary structure assignment," *Proteins*, vol. 23, pp. 566-79, 1995.
[6] P. Y. Chou and G. D. Fasman, "Prediction of protein conformation," *Biochemistry*, vol. 13, pp. 222-45, 1974.
[7] J. Garnier, D. J. Osguthorpe, and B. Robson, "Analysis of the accuracy and implications of simple methods for predicting the secondary structure of globular proteins," *J Mol Biol*, vol. 120, pp. 97-120, 1978.
[8] J. D. Hirst and M. J. Sternberg, "Prediction of structural and functional features of protein and nucleic acid sequences by artificial neural networks," *Biochemistry*, vol. 31, pp. 7211-8, 1992.
[9] D. E. Rumelhart and J. L. McClelland, *Parallel and Distributed Processing*, 1986.
[10] T. J. Sejnowski and N. Qian, "Predicting the Secondary Structure of Globular Proteins Using Neural Network Models," *Journal of Molecular Biology*, vol. 202, pp. 865-884, 1988.
[11] L. H. Holley and M. Karplus, "Protein Secondary Structure Prediction with a Neural Network," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 86, pp. 152-156, 1989.
[12] D. G. Kneller, F. E. Cohen, and R. Langridge, "Improvements in Protein Secondary Structure Prediction by An Enhanced Neural Network," *Journal of Molecular Biology*, vol. 214, pp. 171-182, 1990.
[13] J. M. CHANDONIA and M. KARPLUS, "Neural networks for secondary structure and structural class predictions," *Protein Sci*, vol. 4, pp. 275-285, 1995.
[14] J. M. Chandonia and M. Karplus, "New methods for accurate prediction of protein secondary structure," *Proteins*, vol. 35, pp. 293-306, 1999.
[15] J. A. Cuff, M. E. Clamp, A. S. Siddiqui, M. Finlay, and G. J. Barton, "JPred: a consensus secondary structure prediction server," *Bioinformatics*, vol. 14, pp. 892-3, 1998.
[16] B. Rost, "PHD: predicting one-dimensional protein structure by profile-based neural networks," *Methods in Enzymology*, vol. 266, pp. 525-39, 1996.
[17] M. Riedmiller and H. Braum, "A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm," *IEEE International Conference on Neural Networks*, vol. 1, pp. 586-591, 1993.
[18] T. Kohonen, "The self-organizing map," *Neurocomputing*, vol. 21, pp. 1-6, 1998.
[19] W. Xu and D. P. Miranker, "A metric model of amino acid substitution," *Bioinformatics*, vol. 20, pp. 1214-1221, 2004.
[20] T. Kohonen, *Self-organizing maps*, 3rd ed. Berlin ; New York: Springer, 2001.
[21] C. Liébecq, "Biochemical Nomenclature and Related Documents: a compendium," 2nd ed. London, Chapel Hill: Portland Press, 1992, pp. 347.
[22] R. D. Reed and R. J. Marks, *Neural smithing : supervised learning in feedforward artificial neural networks*. Cambridge, Mass.: The MIT Press, 1999.
[23] S. J. Russell and P. Norvig, *Artificial intelligence : a modern approach*. Englewood Cliffs, N.J.: Prentice Hall, 1995.

[24] Y. Lee and R. P. Lippmann, "Practical characteristics of neural network and conventional pattern classifiers on artificial and speech problems.," *Advances in Neural Information Processing Systems*, vol. 2, pp. 168-177, 1990.

[25] J. L. McClelland and D. E. Rumelhart, *Explorations in Parallel Distributed Processing*: MIT Press, 1988.

[26] J. A. Cuff and G. J. Barton, "Application of Sequence Alignment Profiles to Improve Protein Secondary Structure Prediction," *Proteins: Structure, Function and Genetics*, vol. 40, pp. 502-511, 2000.

[27] J. Moult, K. Fidelis, A. Zemla, and T. Hubbard, "Critical assessment of methods of protein structure prediction (CASP): round IV," *Proteins*, vol. Suppl 5, pp. 2-7, 2001.

[28] J. Moult, K. Fidelis, A. Zemla, and T. Hubbard, "Critical assessment of methods of protein structure prediction (CASP)-round V," *Proteins*, vol. 53 Suppl 6, pp. 334-9, 2003.

[29] J. Moult, T. Hubbard, S. H. Bryant, K. Fidelis, and J. T. Pedersen, "Critical assessment of methods of protein structure prediction (CASP): round II," *Proteins*, vol. Suppl 1, pp. 2-6, 1997.

[30] J. Moult, T. Hubbard, K. Fidelis, and J. T. Pedersen, "Critical assessment of methods of protein structure prediction (CASP): round III," *Proteins*, vol. Suppl 3, pp. 2-6, 1999.

[31] J. A. Cuff and G. J. Barton, "Evaluation and improvement of multiple sequence methods for protein secondary structure prediction," *Proteins*, vol. 34, pp. 508-19, 1999.

[32] B. Rost, C. Sander, and R. Schneider, "Redefining the goals of protein secondary structure prediction," *J Mol Biol*, vol. 235, pp. 13-26, 1994.

[33] A. Zemla, C. Venclovas, K. Fidelis, and B. Rost, "A modified definition of Sov, a segment-based measure for protein secondary structure prediction assessment," *Proteins*, vol. 34, pp. 220-3, 1999.

[34] B. Rost and C. Sander, "Prediction of protein secondary structure at better than 70% accuracy," *J Mol Biol*, vol. 232, pp. 584-99, 1993.

[35] B. Rost and C. Sander, "Combining evolutionary information and neural networks to predict protein secondary structure," *Proteins*, vol. 19, pp. 55-72, 1994.

# Appendix – Amino Acid Encodings

The amino acid encodings referred to in this thesis are listed below.

## Binary

| Amino Acid Code | Encoding |
|---|---|
| - | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 |
| A | 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 |
| B | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 |
| C | 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 |
| D | 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 |
| E | 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 |
| F | 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 |
| G | 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 |
| H | 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 |
| I | 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 |
| K | 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 |
| L | 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 |
| M | 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 |
| N | 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 |
| P | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 |
| Q | 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 |
| R | 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 |
| S | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 |
| T | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 |
| U | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 |
| V | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 |
| W | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 |
| X | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 |
| Y | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 |
| Z | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 |

## Scaled

| Amino Acid Code | Encoding |
|---|---|
| - | 0.0 |
| A | 0.605972 |
| B | 0.540875 |

| | |
|---|---|
| C | 0.344405 |
| D | 0.535232 |
| E | 0.458446 |
| F | 0.773517 |
| G | 0.579425 |
| H | 0.495398 |
| I | 0.689636 |
| K | 0.557823 |
| L | 0.712677 |
| M | 0.672961 |
| N | 0.56799 |
| P | 0.517885 |
| Q | 0.504313 |
| R | 0.472478 |
| S | 0.591856 |
| T | 0.625893 |
| U | 0.1 |
| V | 0.656455 |
| W | 0.9 |
| X | 0.403483 |
| Y | 0.788306 |
| Z | 0.480359 |

## Scaled2

The relative order of encoding is preserved from the scaled encoding but actual codes are uniformly spaced from 0 to 1.

| Amino Acid Code | Encoding |
|---|---|
| - | 0.0 |
| A | 0.64 |
| B | 0.44 |
| C | 0.08 |
| D | 0.4 |
| E | 0.16 |
| F | 0.88 |
| G | 0.56 |
| H | 0.28 |
| I | 0.8 |
| K | 0.48 |
| L | 0.84 |

| | |
|---|---|
| M | 0.76 |
| N | 0.52 |
| P | 0.36 |
| Q | 0.32 |
| R | 0.2 |
| S | 0.6 |
| T | 0.68 |
| U | 0.04 |
| V | 0.72 |
| W | 0.96 |
| X | 0.12 |
| Y | 0.92 |
| Z | 0.24 |

Arbitrary

| Amino Acid Code | Encoding |
|---|---|
| - | 0.0 |
| A | 0.04 |
| B | 0.84 |
| C | 0.2 |
| D | 0.16 |
| E | 0.28 |
| F | 0.56 |
| G | 0.32 |
| H | 0.36 |
| I | 0.4 |
| K | 0.48 |
| L | 0.44 |
| M | 0.52 |
| N | 0.12 |
| P | 0.6 |
| Q | 0.24 |
| R | 0.08 |
| S | 0.64 |
| T | 0.68 |
| U | 0.88 |
| V | 0.8 |
| W | 0.72 |
| X | 0.96 |
| Y | 0.76 |

| | |
|---|---|
| Z | 0.92 |