Research Article

On the Variability of Neural Network Classification Measures in the Protein Secondary Structure Prediction Problem

Eric Sakk and Ayanna Alexander

Department of Computer Science, Morgan State University, Baltimore, MD 21251, USA

Correspondence should be addressed to Eric Sakk; eric.sakk@morgan.edu

Received 27 April 2012; Revised 19 November 2012; Accepted 27 November 2012

Academic Editor: Cheng-Jian Lin

Copyright © 2013 E. Sakk and A. Alexander. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

We revisit the protein secondary structure prediction problem using linear and backpropagation neural network architectures commonly applied in the literature. In this context, neural network mappings are constructed between protein training set sequences and their assigned structure classes in order to analyze the class membership of test data and associated measures of significance. We present numerical results demonstrating that classifier performance measures can vary significantly depending upon the classifier architecture and the structure class encoding technique. Furthermore, an analytic formulation is introduced in order to substantiate the observed numerical data. Finally, we analyze and discuss the ability of the neural network to accurately model fundamental attributes of protein secondary structure.

1. Introduction

The protein secondary structure prediction problem can be phrased as a supervised pattern recognition problem [1–5] for which training data is readily available from reliable databases such as the Protein Data Bank (PDB) or CB513 [6]. Based upon training examples, subsequences derived from primary sequences are encoded based upon a discrete set of classes. For instance, three class encodings are commonly applied in the literature in order to numerically represent the secondary structure set (alpha helix (H), beta sheet (E), coil (C)) [7–11]. By applying a pattern recognition approach, subsequences of unknown classification can then be tested to determine the structure class to which they belong. Phrased in this way, backpropagation neural networks [7, 12–14], and variations on the neural network theme [8, 10, 11, 15–18] have been applied to the secondary structure prediction problem with varied success. Furthermore, many tools currently applying hybrid methodologies such as PredictProtein [19, 20], JPRED [8, 17, 21], SCRATCH [22, 23] and PSIPRED [24, 25] rely on the neural network paradigm as part of their prediction scheme.

One of the main reasons for applying the neural network approach in the first place is that they tend to be good universal

approximators [26–30] and, theoretically, have the potential to create secondary structure models. In other words, after a given network architecture has been chosen and presented with a robust set of examples, the optimal parameters associated with the trained network, in principle, define an explicit function that can map a given protein sequence to its associated secondary structure. If the structure predicted by the network function is generally correct and consistent for an arbitrary input sequence not contained in the training set, one must be left to conclude that the neural network has accurately modeled some fundamental set of attributes that define the properties of protein secondary structure. Under these circumstances, one should then be able to extract information from the trained neural network model parameters; thus, leading to a solution to the secondary structure prediction problem as well as a parametric understanding of the underlying basis for secondary structure.

The purpose of this work is to revisit the application of neural networks to the protein secondary structure prediction problem. In this setting, we consider the commonly encountered case where three structure classes (alpha helix (H), beta sheet (E), and coil (C)) are used to classify a given protein subsequence. Given the same set of input

training sequences, we demonstrate that, for the backpropagation neural network architecture, classification results and associated confidence measures can vary when two equally valid encoding schemes are employed to numerically represent the three structure classes (i.e., the “target encoding scheme”). Such a result goes against the intuition that the physical nature of the secondary structure property should be independent of the target encoding scheme chosen.

The contribution of this work is not to demonstrate improvements over existing techniques. The hybrid techniques outlined above have been demonstrated to outperform neural networks when used alone. Instead, we focus our attention on the ability of the neural network model-based approach to accurately characterize fundamental attributes of protein secondary structure given that certain models presented within this work are demonstrated to yield variable results. Specifically, in this work, we present

- (1) numerical results demonstrating how secondary structure classification results can vary as function of classifier architecture and parameter choices;
- (2) an analytic formulation in order to explain under what circumstances classification variability can arise;
- (3) an outline of specific challenges associated with the neural network model-based approach outlined above.

The conclusions reported here are relevant because they bring into discussion a body of literature that has purported to offer a viable path to the solution of the secondary structure prediction problem. Section 3 describes the methods applied in this work examine the total number that retained their classification using. In particular, this section provides details concerning the encoding of the protein sequence data (Section 3.1), the encoding of the structure classes (Section 3.2) as well as the neural network architectures (Sections 3.3-3.4), and the classifier performance measures (Section 3.5) applied in this work. Section 4 then presents results from numerical secondary structure classification experiments. Section 5 presents an analytic formulation for the linear network and the backpropagation network described in Section 3 in order to explain the numerical results given in Section 4.

2. Notation for the Supervised Classification Problem

In the supervised classification problem [1, 2], it is assumed that a training set consists of N training pairs:

$$\{(X_1, Y_1), (X_2, Y_2), \dots, (X_N, Y_N)\}, \quad (1)$$

where $X_i \in R^n$ are n -dimensional input column vectors and $Y_i \in R^p$ are p -dimensional output column vectors. The goal of the supervised classifier approach is to ensure that the desired response to a given input vector X_i of dimension n from the training set is the p -dimensional output vector Y_i . Furthermore, when the training data can be partitioned into

M distinct classes, a set $\mathcal{E} \equiv \{e_1, \dots, e_j, \dots, e_M\}$ of target p -dimensional column vectors $e_j \in R^p$ are chosen to encode (i.e., mathematically represent) each class for $j = 1, \dots, M$. Under these circumstances, each output training vector Y_i is derived from the set $\mathcal{E}$. Based upon this discussion, we summarize the use of following symbols:

- (i) n is the dimension of a classifier input vector;
- (ii) p is the dimension of a classifier output vector;
- (iii) M is the number of discrete classes for the classification problem;
- (iv) N is the number of training pairs for the supervised classification problem.

3. Methods

In order to apply the neural network paradigm, two numerical issues must be addressed. First, since the input data comes in the form of an amino acid sequence, Section 3.1 discusses a simple encoding scheme for converting the amino acid alphabet into a usable numerical form. Second, for this work, our secondary structure target alphabet consists of elements from the set $\{E, H, C\} = \{\text{Beta Sheet}, \text{AlphaHelix}, \text{Coil}\}$. Hence, an encoding scheme must also be chosen for representing the neural network classifier output. Section 3.2 discusses two approaches to encoding the output in fine detail because it is critical to the main point of this paper. Specifically, we choose two different target vector encoding schemes that can be related by a simple mathematical relationship. Such an approach will allow us to compare classifier performance measures based upon the *target* vector encoding; in addition, it will facilitate the analytic formulation presented in Section 5. Finally, Sections 3.3–3.5 review the neural network architectures and the specific classifier performance measures employed in this work. Section 6 then concludes with some final observations regarding the neural network model-based approach to the protein secondary structure prediction problem.

3.1. Encoding of Protein Sequence Input Data. For the numerical experiments, the training set was constructed using one hundred protein sequences randomly chosen from the CB513 database [6] available through the JPRED secondary structure prediction engine [21]. Furthermore, we employ a moving window of length 17 to each protein sequence where, in order to avoid protein terminal effects, the first and last 50 amino acids are omitted from the analysis. The secondary structure classification of the central residue is then assigned to each window of 17 amino acids. For the one hundred sequences analyzed, a total of 12000 windows of length 17 were extracted. The window size value of 17 was chosen based upon the assumption that the eight closest neighboring residues will have the greatest influence on the secondary structure conformation of the central residue. This assumption is consistent with similar approaches reported in the literature [7, 12–14].

To encode the input amino acid sequences of length 17, we employ sparse orthogonal encoding [31] which maps symbols

from a given sequence alphabet onto a set of orthogonal vectors. Specifically, for an alphabet containing K symbols, a unique K -dimensional unit vector is assigned to each symbol; furthermore, the k th unit vector is one at the k th position and is zero at all other positions. Hence, if all training sequences and unknown test sequences are of uniform length L , an encoded input vector will be of dimension n where $n = LK$. In our case, $K = 20$ and $L = 17$; hence, the dimension of any given input vector is $n = 340$.

The above input vector encoding technique is commonly applied in the bioinformatics and secondary structure prediction literature [7, 15]. While many different and superior approaches to this phase of the machine learning problem have been suggested [3–5], we have chosen orthogonal encoding because of its simplicity and the fact that the results of this work do not depend upon the input encoding scheme. Instead, our work specifically focuses on potential neural network classifier variabilities induced by choice of the target vector encoding scheme.

3.2. Target Vector Encoding. Analytically characterizing the invariance of classifier performance measures clearly involves first establishing a relationship between different sets of target vectors $\mathcal{E}$ and $\tilde{\mathcal{E}}$. As a means of making the invariance formulation presented in this paper more tractable, we assume that two alternative sets of target vectors can be related via an affine transformation involving a translation, T , a rigid rotation, R , where R is an orthogonal matrix and a scale factor, a ,

$$\tilde{\Gamma} \equiv aR(\Gamma - T), \quad (2)$$

where

$$\Gamma \equiv [e_1 \ e_2 \ \dots \ e_M], \quad \tilde{\Gamma} \equiv [\tilde{e}_1 \ \tilde{e}_2 \ \dots \ \tilde{e}_M], \quad (3)$$

$$T \equiv [t_1 \ t_2 \ \dots \ t_M]$$

is a matrix of translation column vectors applied to each target vector. Many target vector choices regularly applied in the literature can be related via the transformation in (2). For instance, two equally valid and commonly applied encoding schemes for the three class problem are orthogonal encoding [31] where

$$H \rightarrow \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad E \rightarrow \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad C \rightarrow \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad (4)$$

and

$$H \rightarrow \begin{bmatrix} -\frac{1}{2} \\ \frac{\sqrt{3}}{2} \\ 0 \end{bmatrix}, \quad E \rightarrow \begin{bmatrix} -\frac{1}{2} \\ -\frac{\sqrt{3}}{2} \\ 0 \end{bmatrix}, \quad C \rightarrow \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad (5)$$

where class encodings are chosen on the vertices of a triangle in a two-dimensional plane [14]. It turns out that (4) and (5) can be phrased in terms of (2) [32]; hence, the numerical

results presented in this work will apply this set of encodings. More precisely, the secondary structure classification associated with a given input vector is encoded using (4) and (5) (hence, $p = 3$). The set of target vectors $\mathcal{E}$ is derived from (4) and the set of target vectors $\tilde{\mathcal{E}}$ is derived from (5). Both the linear and the backpropagation networks are tested first by training using $\mathcal{E}$ and then comparing classifier performance with their counterparts trained using $\tilde{\mathcal{E}}$. In all numerical experiments, MATLAB has been used for simulating and testing these networks.

3.3. The Linear Network. When the supervised classifier model in (1) assumes an affine relationship between the input and output data sets (as in the case of multiple linear regression), matrices of the form

$$X \equiv [X_1 \ \dots \ X_N], \quad (6)$$

$$Y \equiv [Y_1 \ \dots \ Y_N] \quad (7)$$

are generally introduced. Specifically, the linear network seeks to determine a $p \times n$ matrix β of coefficients and a constant p -dimensional column vector c such that the i th output vector Y_i in the training set can be approximated by

$$\beta X_i + c. \quad (8)$$

Given this model, we can form a weight matrix of unknown coefficients

$$W \equiv [\beta \ c] \quad (9)$$

that, ideally, will map each input training vector into the corresponding output training vector. If the bottom row of the input data matrix X is appended with a row of ones leading to the $(n+1) \times N$ matrix

$$A \equiv \begin{bmatrix} X_1 & \dots & X_N \\ 1 & \dots & 1 \end{bmatrix}, \quad (10)$$

in matrix form, the goal is then to find a $p \times n+1$ weight matrix W that minimizes the sum squared-error

$$E^{\text{linear}} \equiv \sum_{i=1}^N (WA_i - Y_i)^T (WA_i - Y_i) \quad (11)$$

over the set of data pairs

$$A_i \equiv \begin{bmatrix} X_i \\ 1 \end{bmatrix} \quad (12)$$

by satisfying the first derivative condition $\partial E^{\text{linear}} / \partial W = 0$. The least squares solution to this problem is found via the pseudoinverse $A^\dagger$ [33] where

$$W = YA^\dagger. \quad (13)$$

Once the optimal set of weights has been computed, the network response to an unknown $n \times 1$ input vector x can be determined by defining the $(n+1) \times 1$ vector

$$z \equiv [x \ 1]^T \quad (14)$$

and calculating

$$f(z) = Wz, \quad (15)$$

where $f(z)$ is a $p \times 1$ column vector.

3.4. The Backpropagation Network. Given an input vector $x \in R^n$, the model $f : R^n \rightarrow R^p$ for a backpropagation neural network with a single hidden layer consisting of m nodes is described as

$$f(x) = \begin{bmatrix} f_1(x) \\ f_2(x) \\ \vdots \\ f_p(x) \end{bmatrix} = \alpha \sigma(Wx - \tau), \quad (16)$$

where $\alpha \in R^{p \times m}$, $W \in R^{m \times n}$, and $\tau \in R^m$ define the set of network weights $\mathcal{W} = \{W, \tau, \alpha\}$ and $\sigma : R^m \rightarrow R^p$ is a “sigmoidal” function that is bounded and monotonically increasing. To perform supervised training, in a manner similar to the linear network, $\mathcal{W}$ is determined by minimizing the objective function:

$$E^{bp} = \sum_{i=1}^N (f(X_i) - Y_i)^T (f(X_i) - Y_i) \quad (17)$$

given the training data defined in (1). Since $f(X_i)$ is no longer linear, numerical techniques such as the gradient descent algorithm and variations thereof are relied upon to compute the set $\mathcal{W}$ that satisfies the first derivative condition $\partial E^{bp} / \partial \mathcal{W} = 0$.

Consider the following definitions:

$$\delta F_k \equiv f(X_k) - Y_k = \begin{bmatrix} f_1(X_k) - (Y_k)_1 \\ f_2(X_k) - (Y_k)_2 \\ \vdots \\ f_p(X_k) - (Y_k)_p \end{bmatrix},$$

$$S_k \equiv \begin{bmatrix} \sigma \left(\sum_{j=1}^n W_{1j}(X_k)_j - \tau_1 \right) \\ \sigma \left(\sum_{j=1}^n W_{2j}(X_k)_j - \tau_2 \right) \\ \vdots \\ \sigma \left(\sum_{j=1}^n W_{mj}(X_k)_j - \tau_m \right) \end{bmatrix}, \quad (18)$$

$$S'_k \equiv \begin{bmatrix} \sigma' \left(\sum_{j=1}^n W_{1j}(X_k)_j - \tau_1 \right) \\ \sigma' \left(\sum_{j=1}^n W_{2j}(X_k)_j - \tau_2 \right) \\ \vdots \\ \sigma' \left(\sum_{j=1}^n W_{mj}(X_k)_j - \tau_m \right) \end{bmatrix},$$

where $\sigma'(z) = d\sigma/dz$. The first derivative conditions for the network weights prescribed by (16) and (17) can then be written in matrix form as follows:

$$\frac{\partial E^{bp}}{\partial \alpha} = \sum_{k=1}^N \delta F_k S_k^T = 0, \quad (19)$$

(where “ T ” denotes the matrix transpose),

$$\frac{\partial E^{bp}}{\partial \tau} = - \sum_{k=1}^N \text{diag}(S'_k) \alpha^T \delta F_k^T = 0, \quad (20)$$

$$\frac{\partial E^{bp}}{\partial W} = \sum_{k=1}^N \text{diag}(S'_k) \alpha^T \delta F_k^T X_k^T = 0,$$

where $\text{diag}(S'_k)$ is a square diagonal matrix such that the diagonal entries consist of components from the vector S'_k .

3.5. Classification Measures. After a given classifier is trained, when presented with an input vector $x \in R^n$ of unknown classification, it will respond with an output $f(x) \in R^p$. The associated class membership $C(x)$ is then often determined by applying a minimum distance criterion:

$$C(x) \equiv \arg \min_i d(f(x), e_i), \quad (21)$$

where the target vector e_i that is closest to $f(x)$ implies the class. Furthermore, when characterizing the performance of a pattern classifier, one often presents a set S of test vectors and analyzes the associated output. In addition to determining the class membership, it is also possible to *rank* the distance between a specific target vector and the classifier response to S . In this case a similar distance criterion can be applied in order to rank an input vector $x \in S$ with respect to class $J \in \{1, \dots, M\}$,

$$\rho_J(x) = d(f(x), e_J). \quad (22)$$

For the purposes of this work, (15) facilitates the determination of the class membership and ranking with respect to class J for the linear network. Similarly, assuming a set of weights $\mathcal{W} = \{W, \tau, \alpha\}$ for a trained backpropagation network, (21) and (22) would be applied using (16).

It is well established that, in the case of a normally distributed data, the classification measures presented above minimize the probability of a classification error and are directly related to the statistical significance of a classification decision [1]. Given the neural network as the supervised classification technique and two distinct choices for the set of target vectors $\mathcal{E}$ and $\tilde{\mathcal{E}}$, we demonstrate, in certain instances, that the classification and ranking results do not remain *invariant* such that $\tilde{C}(x) \neq C(x)$ and $\tilde{\rho}_J(x) \neq \rho_J(x)$ for any input vector x .

4. Noninvariance of Secondary Structure Predictions

In this section, we numerically demonstrate that, when different target vector encodings are applied, neural network classifier measures outlined above, in certain cases, are observed to vary widely. For each neural network architecture under consideration, an analytic formulation is then presented in Section 5 in order to explain the observed numerical data.

As mentioned in Section 3.2, numerical experiments are performed first by training using $\mathcal{E}$ and then comparing

TABLE 1: Ranking results for the linear network where i represents the test vector index and $\rho_1(x)$ and $\bar{\rho}_1(x)$ represent the distance with respect to the helix class vectors e_1 and $\bar{e}_1$. Out of 1800 vectors tested, vectors referred to in this table were ranked from 1 to 20.

$i_{\mathcal{E}}$	$\rho_1(x)$	$i_{\bar{\mathcal{E}}}$	$\bar{\rho}_1(x)$
1205	0.0780	1205	0.1170
42	0.0867	42	0.1300
1031	0.0976	1031	0.1464
1773	0.1113	1773	0.1670
598	0.1238	598	0.1857
1761	0.1267	1761	0.1900
862	0.1354	862	0.2031
1073	0.1409	1073	0.2114
277	0.1459	277	0.2188
115	0.1540	115	0.2309
1505	0.1821	1505	0.2731
392	0.1839	392	0.2759
1421	0.1904	1421	0.2856
147	0.2001	147	0.3001
990	0.2044	990	0.3066
1457	0.2127	1457	0.3191
1288	0.2150	1288	0.3225
352	0.2160	352	0.3239
1232	0.2198	1232	0.3297
280	0.2311	280	0.3466

classifier performance with their counterparts trained using $\bar{\mathcal{E}}$. Multiple cross-validation trials are required in order to prevent potential dependency of the evaluated accuracy on the particular training or test sets chosen [7, 15]. In this work, we apply a hold-n-out strategy similar to that of [14] using 85% of the 12000 encoded sequences as training data (i.e., $N = 10200$) and 15% as test data to validate the classification results. Recognition rates for both the linear and backpropagation rates using either set of target vector encodings were approximately 65% which is typical of this genre of classifiers that have applied similar encoding methodologies [7, 12–14]. Although these aggregate values remain consistent, using (21) and (22) we now present data demonstrating that, while class membership and ranking remain invariant for the linear network, these measures of performance vary considerably for the backpropagation network which was trained with, $m = 17$, seventeen hidden nodes and a mean squared training error less than 0.2. Ranking results from a representative test for the linear and backpropagation networks are presented for the top 20 ranked vectors in Tables 1 and 3. Class membership data are presented in Tables 2 and 4. Observe that, for the linear network, indices for the top 20 ranked vectors remain invariant indicating ranking invariance; in addition, no change in class membership is observed. On the other hand, Tables 3 and 4 clearly indicate a lack of consistency when considering the ranking and class membership of test vectors. A particularly troubling observation is that very few vectors ranked in the top 20 with respect to e_1 were ranked in the top 20 with respect to $\bar{e}_1$. Furthermore, Table 4 indicates

TABLE 2: Class membership results for the linear network. For each class, the total number of vectors classified using $\mathcal{E}$ is analyzed to examine the total number that retained their classification using $\bar{\mathcal{E}}$.

Class	$\mathcal{E}$	$\bar{\mathcal{E}}$	% change
H	202	202	0
E	621	621	0
C	977	977	0

that the class membership of a substantial number of test vectors changed when an alternative set of target vectors was employed. The data also indicates that the greatest change in class membership took place for alpha helical sequences; thus implying that there is substantial disagreement over the modeling of this secondary structure element by the backpropagation network due to a simple transformation of the target vectors.

5. Analysis

The results in Section 4 clearly show that while the pattern recognition results for the linear network remain invariant under a change in target vectors, those for the backpropagation network do not. In this section, we present analytic results in order to clearly explain and understand why these two techniques lead to different conclusions.

5.1. Invariance Formulation. Let us begin by considering two definitions.

Definition 1. Given two sets of target vectors $\mathcal{E}$ and $\bar{\mathcal{E}}$, the class membership is invariant under a transformation of the target vectors if, for any input vector x ,

$$d(\tilde{f}(x), \bar{e}_i) > d(\tilde{f}(x), \bar{e}_j) \implies d(f(x), e_i) > d(f(x), e_j), \quad (23)$$

where $\tilde{f}(x)$ is the output of the classifier with target vectors $\{\bar{e}_j\}_{j=1}^M$.

Definition 2. Given two sets of target vectors $\mathcal{E}$ and $\bar{\mathcal{E}}$, the ranking with respect to a specific class J is invariant under a transformation of the target vectors if, for any input vectors x_1 and x_2 ,

$$d(\tilde{f}(x_1), \bar{e}_j) > d(\tilde{f}(x_2), \bar{e}_j) \implies d(f(x_1), e_j) > d(f(x_2), e_j). \quad (24)$$

Based upon these definitions, the following has been established [32].

Proposition 3. Given two sets of target vectors $\mathcal{E}$ and $\bar{\mathcal{E}}$, if the ranking is invariant, then the class membership of an arbitrary input vector x will remain invariant.

In the analysis presented, the strategy for characterizing neural network performance depends upon the data from the previous section. For the linear network, since both ranking

TABLE 3: Ranking results for the backpropagation network where i represents the test vector index and $\rho_1(x)$ and $\tilde{\rho}_1(x)$ represent the distance with respect to the helix class vectors e_1 and $\tilde{e}_1$. Out of 1800 vectors tested, vectors referred to in this table were ranked from 1 to 20.

$i_{\mathcal{E}}$	$\rho_1(x)$	$i_{\tilde{\mathcal{E}}}$	$\tilde{\rho}_1(x)$
817	0.0107	926	0.0101
887	0.0231	1604	0.0130
264	0.0405	887	0.0209
1183	0.0711	1145	0.0214
684	0.0727	461	0.0232
623	0.0874	583	0.0329
911	0.0891	1086	0.0339
1382	0.0917	1382	0.0478
1610	0.0939	413	0.0489
551	0.1060	225	0.0608
1042	0.1150	438	0.0609
924	0.1322	911	0.0613
727	0.1339	207	0.0774
438	0.1356	559	0.0885
577	0.1363	481	0.0945
896	0.1500	1548	0.0947
175	0.1513	962	0.0968
1138	0.1549	85	0.1012
583	0.1581	195	0.1111
559	0.1655	9	0.1167

TABLE 4: Class membership results for the backpropagation network. For each class, the total number of vectors classified using $\mathcal{E}$ is analyzed to examine the total number that retained their classification using $\tilde{\mathcal{E}}$.

Class	$\mathcal{E}$	$\tilde{\mathcal{E}}$	% change
H	225	142	36.9
E	581	476	18.1
C	994	878	11.7

and classification were observed to remain invariant, it is more sensible to characterize the invariance of this network using Definition 2. Then, based upon Proposition 3, class membership invariance naturally follows. On the other hand, to explain the noninvariance of both class membership and ranking observed in the backpropagation network, the analysis is facilitated by considering Definition 1. The noninvariance of ranking then naturally follows from Proposition 3.

5.1.1. Invariance Analysis for the Linear Network. When the target vectors are subjected to the transformation defined in (2), the network output can be expressed as

$$\tilde{f}(z) = \tilde{W}z = \tilde{Y}A^\dagger z = aR(Y - T')A^\dagger z, \quad (25)$$

where T' is derived from T such that the translation vector t_i associated with $e_i \in \mathcal{E}$ is appropriately aligned with the

correct target vector in matrix Y . In other words, when the output data matrix in (7) is of the form

$$Y = [e_{i_1} \ e_{i_2} \ \dots \ e_{i_N}] \quad (26)$$

then

$$T' = [t_{i_1} \ t_{i_2} \ \dots \ t_{i_N}], \quad (27)$$

where $i_j \in \{1, \dots, M\}$ for $j = 1, \dots, N$. Given this network, the following result is applicable [32].

Proposition 4. *If*

- (i) *the number of training observations N exceeds the vector dimension n ;*
- (ii) *the rows of the matrix A are linearly independent;*
- (iii) *$\mathcal{E}$ and $\tilde{\mathcal{E}}$ are related according to (2);*
- (iv) *for some $t \in R^p$, $t_i = t$ for all $i = 1, \dots, M$ in (2);*

then the ranking and, hence, the class membership for the linear network will remain invariant.

In other words, if the columns of the matrix T' in (25) are all equal, then using (15) and (25) will result in, (23) being satisfied. The above result is applicable to the presented numerical data with $n = 340$ and $N = 10200$; hence, ranking and class membership invariances are corroborated by the data in Tables 1 and 2.

5.1.2. Invariance Analysis for the Backpropagation Network.

In this section, we seek to characterize the noninvariance observed in class membership using the backpropagation network. If the class membership varies due to a change in target vectors, then this variation should be quantifiable by characterizing the boundary separating two respective classes. The decision boundary between class i and class j is defined by points x such that

$$\|f(x) - e_i\|^2 = \|f(x) - e_j\|^2, \quad (28)$$

where $i, j \in \{1, \dots, M\}$, $i \neq j$ and $f(x)$ is the classifier output. Under these circumstances, if an ℓ_2 norm is applied in (21),

$$(f(x) - e_i)^T (f(x) - e_i) = (f(x) - e_j)^T (f(x) - e_j). \quad (29)$$

The solution set to this equation consists of all x such that $f(x)$ is equidistant from e_i and e_j where, for the purposes of this section, $f(x)$ is defined by (16). Expanding terms on both sides of this equation leads to the condition

$$(e_i - e_j)^T f(x) = -\frac{1}{2} (e_j^T e_j - e_i^T e_i). \quad (30)$$

If the class membership of a representative vector is to remain invariant under a change of target vectors, this same set of points must also satisfy

$$(\tilde{e}_i - \tilde{e}_j)^T \tilde{f}(x) = -\frac{1}{2} (\tilde{e}_j^T \tilde{e}_j - \tilde{e}_i^T \tilde{e}_i). \quad (31)$$

Assuming that two networks have been trained using two different sets of target vectors $\mathcal{E}$ and $\tilde{\mathcal{E}}$, the set of weights $\{W, \tau, \alpha\}$ and $\{\tilde{W}, \tilde{\tau}, \tilde{\alpha}\}$ determines the network output in (16). Without loss of generality, we consider the case where all target vectors are normalized to a value of one such that $\|e_i\|^2 = 1$ and $\|\tilde{e}_i\|^2 = 1$ for $i = 1, \dots, M$. In this case, the conditions in (30) and (31) become

$$(e_i - e_j)^T \alpha \sigma(Wx - \tau) = 0, \quad (32)$$

$$(\tilde{e}_i - \tilde{e}_j)^T \tilde{\alpha} \sigma(\tilde{W}x - \tilde{\tau}) = 0. \quad (33)$$

We first consider a special case where the target vectors are related according to $\tilde{T} = aR\Gamma$ with $T = 0$ in (2). Under these circumstances, if the choice

$$\mathcal{W} \equiv \{\tilde{W}, \tilde{\tau}, \tilde{\alpha}\} = \{W, \tau, aR\alpha\} \quad (34)$$

is made, it should be clear that, since $R^T = R^{-1}$ and $\tilde{Y}_i = aRY_i$,

$$\begin{aligned} \tilde{E}^{bp} &= \sum_{i=1}^N (\tilde{f}(X_i) - \tilde{Y}_i)^T (\tilde{f}(X_i) - \tilde{Y}_i) \\ &= a^2 \sum_{i=1}^N (Rf(X_i) - RY_i)^T (Rf(X_i) - RY_i) \\ &= a^2 \sum_{i=1}^N (f(X_i) - Y_i)^T R^T R (f(X_i) - Y_i) \\ &= a^2 E^{bp} \end{aligned} \quad (35)$$

is minimized by the condition $\partial \tilde{E}^{bp} / \partial \mathcal{W} = 0$. Another way to see this is to observe that (19), (20) remain invariant for this choice of target vectors and network weights. Hence, we have the following.

Proposition 5. For a specific choice of a and R , if $T = 0$ in (2) and

$$\tilde{\mathcal{W}} = \{\tilde{W}, \tilde{\tau}, \tilde{\alpha}\} = \{W, \tau, aR\alpha\}, \quad (36)$$

then the class membership for the backpropagation network will remain invariant.

Proof. Simply consider (32) and (33) and choose any $\bar{x}$ satisfying

$$(e_i - e_j)^T \alpha \sigma(W\bar{x} - \tau) = 0. \quad (37)$$

It then immediately follows that

$$\begin{aligned} &(\tilde{e}_i - \tilde{e}_j)^T \tilde{\alpha} \sigma(\tilde{W}\bar{x} - \tilde{\tau}) \\ &= (e_i - e_j)^T R^T R \alpha \sigma(W\bar{x} - \tau) \\ &= a^2 (e_i - e_j)^T \alpha \sigma(W\bar{x} - \tau) \\ &= 0. \end{aligned} \quad (38)$$

Therefore, if $\bar{x}$ satisfies (32), then it also satisfies (33) and, hence, is a point on the decision boundary for both networks. $\square$

Intuitively, a scaled, rigid rotation of the target vectors should not affect the decision boundary. However, when the more general transformation of (2) is applied with $T \neq 0$, we now demonstrate that, due to the nonlinearity of σ in (16), no simple relationship exists such that (32) and (33) can simultaneously be satisfied by the same set of points. We first investigate the possibility of establishing an analytic relationship between the set of weights $\{\tilde{W}, \tilde{\tau}, \tilde{\alpha}\}$ and $\{W, \tau, \alpha\}$ for both networks. In other words, we seek, ideally invertible, functions G_α , G_τ , and G_W .

$$\begin{aligned} \tilde{\alpha} &= G_\alpha(W, \tau, \alpha), \\ \tilde{\tau} &= G_\tau(W, \tau, \alpha), \\ \tilde{W} &= G_W(W, \tau, \alpha) \end{aligned} \quad (39)$$

such that the set $\tilde{\mathcal{W}}$ can be transformed into $\mathcal{W}$. If this can be done, then an analytic procedure similar to that presented in the proof of Proposition 5 can be established in order to relate (32) to (33) for the general case. Since (19), (20) define the set $\mathcal{W}$, it is reasonable to rephrase these equations in terms of the objective function $\tilde{E}^{bp}$:

$$\begin{aligned} \frac{\partial \tilde{E}^{bp}}{\partial \tilde{\alpha}} &= \sum_{k=1}^N (\tilde{f}(X_k) - \tilde{Y}_k) \tilde{S}_k^T = 0, \\ \frac{\partial \tilde{E}^{bp}}{\partial \tilde{\tau}} &= - \sum_{k=1}^N \text{diag}(\tilde{S}_k') \tilde{\alpha}^T (\tilde{f}(X_k) - \tilde{Y}_k)^T = 0, \\ \frac{\partial \tilde{E}^{bp}}{\partial \tilde{W}} &= \sum_{k=1}^N \text{diag}(\tilde{S}_k') \tilde{\alpha}^T (\tilde{f}(X_k) - \tilde{Y}_k)^T X_k^T = 0, \end{aligned} \quad (40)$$

where $\tilde{f}(X_k) = \tilde{\alpha} \sigma(\tilde{W}X_k - \tilde{\tau})$ and $\tilde{Y}_k = aR(Y_k - t')$ such that t' is the translation vector associated with the target vector referred to by Y_k . From these equations, it should be clear that no simple analytic relationship exists that will transform $\tilde{\mathcal{W}}$ into $\mathcal{W}$. A numerical algorithm such as gradient descent will, assuming a local minimum actually exists, arrive at some solution for both $\tilde{\mathcal{W}}$ and $\mathcal{W}$. We must therefore be content with the assumed existence of some set of functions defined by (39). Again, let us consider any point $\bar{x}$ on the decision boundary such that

$$(e_i - e_j)^T \alpha \sigma(W\bar{x} - \tau) = 0. \quad (41)$$

Such a point must also simultaneously satisfy

$$\begin{aligned} &(\tilde{e}_i - \tilde{e}_j)^T \tilde{\alpha} \sigma(\tilde{W}\bar{x} - \tilde{\tau}) \\ &= ((aRe_i - t_i) - (aRe_j - t_j))^T G_\alpha(W, \tau, \alpha) \end{aligned}$$

$$\begin{aligned}
& \times \sigma(G_W(W, \tau, \alpha) \bar{x} - G_\tau(W, \tau, \alpha)) \\
& = a((e_i - e_j) - (t_i - t_j))^T R^T G_\alpha(W, \tau, \alpha) \\
& \quad \times \sigma(G_W(W, \tau, \alpha) \bar{x} - G_\tau(W, \tau, \alpha)) \\
& = 0.
\end{aligned} \tag{42}$$

At first glance, a choice such as $G_\alpha(W, \tau, \alpha) = aR\alpha$ and $t_i = t_j$ for $i, j = 1, \dots, M$ (as in Proposition 4) appears to bring us close to a solution. However, the term involving $\sigma(G_W(W, \tau, \alpha) \bar{x} - G_\tau(W, \tau, \alpha))$ is problematic. Although a choice such as $G_W(W, \tau, \alpha) = W$ and $G_\tau(W, \tau, \alpha) = \tau$ would yield the solution to (42), it should be clear that these values would not satisfy (40).

Another way to analyze this problem is to first set $t_i = t_j$ for $i, j = 1, \dots, M$. Then, for any $\bar{x}$ on the decision boundary, from (41) and (42), equate the terms

$$\begin{aligned}
\alpha \sigma(W\bar{x} - \tau) &= kR^T G_\alpha(W, \tau, \alpha) \\
&\quad \times \sigma(G_W(W, \tau, \alpha) \bar{x} - G_\tau(W, \tau, \alpha)),
\end{aligned} \tag{43}$$

where k is some constant and G_α , G_τ , and G_W satisfy (40). Given an arbitrary training set defined by (1), it is highly unlikely that this constraint can be satisfied. One remote scenario might be that the terms $G_W(W, \tau, \alpha) \bar{x} - G_\tau(W, \tau, \alpha)$ and $W\bar{x} - \tau$ are always small. In this case, given a sigmoidal function $\sigma(z)$ that is linear near $z = 0$, a linearized version of (43) could be solved using techniques described in [32]. However, this again is an unlikely set of events given an arbitrary training set. Therefore, given the transformation of (2), we are left to conclude that class membership invariance and, hence, ranking invariance are, in general, not achievable using the backpropagation neural network.

5.2. Discussion. Intuitively, given a reasonable target encoding scheme, one would desire that properties related to protein secondary structure would be independent of the target vectors chosen. However, we have presented numerical data and a theoretical foundation demonstrating that secondary structure classification and confidence measures can vary depending on the type of neural network architecture and target vector encoding scheme employed. Specifically, linear network classification has been demonstrated to remain invariant under a change in the target structure encoding scheme while the backpropagation network has not. As N increases, for the methodology applied in this work, recognition rates remain consistent with those reported in the literature; however, we have observed that adding more training data does not improve the invariance of classification measures for the backpropagation network. This conclusion is corroborated by the analytic formulation presented above.

6. Conclusions

As pointed out in the introduction, one major purpose of the neural network is to create a stable and reliable model that

maps input training data to an output classification with the hope of extracting informative parameters. When methods similar to those in the literature are applied [7, 12–14], we have demonstrated that classifier performance measures can vary considerably. Under these circumstances, parameters derived from a trained network for analytically describing protein secondary structure may not comprise a reliable set for the model-based approach. Furthermore, classifier variability would imply that a stable parametric model has not been derived. It is in some sense paradoxical that the neural network has been applied for structure classification and, yet, associated parameters have not been applied for describing protein secondary structure. The neural network approach to deriving a solution to the protein secondary structure prediction problem therefore requires deeper exploration.

Acknowledgments

This publication was made possible by Grant Number G12RR017581 from the National Center for Research Resources (NCRR), a component of the National Institutes of Health (NIH). The authors would also like to thank the reviewers for their helpful comments.

References

- [1] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2007.
- [2] M. T. Hagan, H. B. Demuth, and M. H. Beale, *Neural Network Design*, PWS Publishing, 1996.
- [3] M. N. Nguyen and J. C. Rajapakse, “Multi-class support vector machines for protein secondary structure prediction,” *Genome Informatics*, vol. 14, pp. 218–227, 2003.
- [4] H. J. Hu, Y. Pan, R. Harrison, and P. C. Tai, “Improved protein secondary structure prediction using support vector machine with a new encoding scheme and an advanced tertiary classifier,” *IEEE Transactions on Nanobioscience*, vol. 3, no. 4, pp. 265–271, 2004.
- [5] W. Zhong, G. Altun, X. Tian, R. Harrison, P. C. Tai, and Y. Pan, “Parallel protein secondary structure prediction schemes using Pthread and OpenMP over hyper-threading technology,” *Journal of Supercomputing*, vol. 41, no. 1, pp. 1–16, 2007.
- [6] J. A. Cuff and G. J. Barton, “Application of enhanced multiple sequence alignment profiles to improve protein secondary structure prediction,” *Proteins*, vol. 40, pp. 502–511, 2000.
- [7] J. M. Chandonia and M. Karplus, “Neural networks for secondary structure and structural class predictions,” *Protein Science*, vol. 4, no. 2, pp. 275–285, 1995.
- [8] J. A. Cuff and G. J. Barton, “Evaluation and improvement of multiple sequence methods for protein secondary structure prediction,” *Proteins*, vol. 34, pp. 508–519, 1999.
- [9] G. E. Crooks and S. E. Brenner, “Protein secondary structure: entropy, correlations and prediction,” *Bioinformatics*, vol. 20, no. 10, pp. 1603–1611, 2004.
- [10] L. H. Wang, J. Liu, and H. B. Zhou, “A comparison of two machine learning methods for protein secondary structure prediction,” in *Proceedings of 2004 International Conference on Machine Learning and Cybernetics*, pp. 2730–2735, chn, August 2004.

- [11] G. Z. Zhang, D. S. Huang, Y. P. Zhu, and Y. X. Li, "Improving protein secondary structure prediction by using the residue conformational classes," *Pattern Recognition Letters*, vol. 26, no. 15, pp. 2346–2352, 2005.
- [12] N. Qian and T. J. Sejnowski, "Predicting the secondary structure of globular proteins using neural network models," *Journal of Molecular Biology*, vol. 202, no. 4, pp. 865–884, 1988.
- [13] L. Howard 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, no. 1, pp. 152–156, 1989.
- [14] J. M. Chandonia and M. Karplus, "The importance of larger data sets for protein secondary structure prediction with neural networks," *Protein Science*, vol. 5, no. 4, pp. 768–774, 1996.
- [15] B. Rost and C. Sander, "Improved prediction of protein secondary structure by use of sequence profiles and neural networks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 90, no. 16, pp. 7558–7562, 1993.
- [16] B. Rost and C. Sander, "Prediction of protein secondary structure at better than 70% accuracy," *Journal of Molecular Biology*, vol. 232, no. 2, pp. 584–599, 1993.
- [17] 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, no. 10, pp. 892–893, 1998.
- [18] S. Hua and Z. Sun, "A novel method of protein secondary structure prediction with high segment overlap measure: support vector machine approach," *Journal of Molecular Biology*, vol. 308, no. 2, pp. 397–407, 2001.
- [19] B. Rost, "PHD: predicting one-dimensional protein structure by profile-based neural networks," *Methods in Enzymology*, vol. 266, pp. 525–539, 1996.
- [20] B. Rost, G. Yachdav, and J. Liu, "The PredictProtein server," *Nucleic Acids Research*, vol. 32, pp. W321–W326, 2004.
- [21] C. Cole, J. D. Barber, and G. J. Barton, "The Jpred 3 secondary structure prediction server," *Nucleic Acids Research*, vol. 36, pp. W197–W201, 2008.
- [22] G. Pollastri, D. Przybylski, B. Rost, and P. Baldi, "Improving the prediction of protein secondary structure in three and eight classes using recurrent neural networks and profiles," *Proteins*, vol. 47, no. 2, pp. 228–235, 2002.
- [23] J. Cheng, A. Z. Randall, M. J. Sweredoski, and P. Baldi, "SCRATCH: a protein structure and structural feature prediction server," *Nucleic Acids Research*, vol. 33, no. 2, pp. W72–W76, 2005.
- [24] D. T. Jones, "Protein secondary structure prediction based on position-specific scoring matrices," *Journal of Molecular Biology*, vol. 292, no. 2, pp. 195–202, 1999.
- [25] K. Bryson, L. J. McGuffin, R. L. Marsden, J. J. Ward, J. S. Sodhi, and D. T. Jones, "Protein structure prediction servers at University College London," *Nucleic Acids Research*, vol. 33, no. 2, pp. W36–W38, 2005.
- [26] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals, and Systems*, vol. 2, no. 4, pp. 303–314, 1989.
- [27] J. Moody and C. J. Darken, "Fast learning in networks of locally tuned processing units," *Neural Computation*, vol. 1, pp. 281–294, 1989.
- [28] T. Poggio and F. Girosi, "Networks for approximation and learning," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1481–1497, 1990.
- [29] D. F. Specht, "Probabilistic neural networks," *Neural Networks*, vol. 3, no. 1, pp. 109–118, 1990.
- [30] P. András, "Orthogonal RBF neural network approximation," *Neural Processing Letters*, vol. 9, no. 2, pp. 141–151, 1999.
- [31] P. Baldi and S. Brunak, *Bioinformatics: The Machine Learning Approach*, MIT Press, 1998.
- [32] E. Sakk, D. J. Schneider, C. R. Myers, and S. W. Cartinhour, "On the selection of target vectors for a class of supervised pattern recognizers," *IEEE Transactions on Neural Networks*, vol. 20, no. 5, pp. 745–757, 2009.
- [33] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, 1989.