PROTEINS
STRUCTURE ■ FUNCTION ■ BIOINFORMATICS

Improving the prediction accuracy of residue solvent accessibility and real-value backbone torsion angles of proteins by guided-learning through a two-layer neural network

Eshel Faraggi,[1,2] Bin Xue,[1,2] and Yaoqi Zhou[1,2]*

[1] Indiana University School of Informatics, Indiana University-Purdue University, Indianapolis, IN 46202
[2] Center for Computational Biology and Bioinformatics, Indiana University School of Medicine, 719 Indiana Ave., Walker Plaza Building Suite 319, Indianapolis, IN 46202, USA

ABSTRACT

This article attempts to increase the prediction accuracy of residue solvent accessibility and real-value backbone torsion angles of proteins through improved learning. Most methods developed for improving the backpropagation algorithm of artificial neural networks are limited to small neural networks. Here, we introduce a guided-learning method suitable for networks of any size. The method employs a part of the weights for guiding and the other part for training and optimization. We demonstrate this technique by predicting residue solvent accessibility and real-value backbone torsion angles of proteins. In this application, the guiding factor is designed to satisfy the intuitive condition that for most residues, the contribution of a residue to the structural properties of another residue is smaller for greater separation in the protein-sequence distance between the two residues. We show that the guided-learning method makes a 2–4% reduction in 10-fold cross-validated mean absolute errors (MAE) for predicting residue solvent accessibility and backbone torsion angles, regardless of the size of database, the number of hidden layers and the size of input windows. This together with introduction of two-layer neural network with a bipolar activation function leads to a new method that has a MAE of 0.11 for residue solvent accessibility, 36° for ψ, and 22° for ϕ. The method is available as a Real-SPINE 3.0 server in http://sparks.informatics. iupui.edu.

Proteins 2009; 74:847–856.

Key words: artificial neural networks; dihedral angles; solvent-accessible surface area; protein structure prediction.

INTRODUCTION

Direct prediction of protein structures from their sequences is challenging. As a result, protein structure prediction is often assisted by predicting one-dimensional structural properties including residue solvent-accessibility (RSA) and backbone torsion angles of proteins. Although the usefulness of predicted RSA values in structure prediction is well established,[1–6] the application of predicted torsion angles is still in its infancy (fold recognition,[7–9] sequence alignment,[10] and secondary structure prediction.[11,12]) However, the latter has the potential to replace or supplement predicted secondary structures[13,14] because torsion angles provide a more detailed description of the backbone structure than three-state secondary structures.

Both residue solvent-accessible surface areas and backbone angles are continuously varying variables because proteins can move freely in a three-dimensional space. Thus, a real-value prediction is preferred over the prediction of a few arbitrarily-defined states. Although several methods for real-value prediction of solvent accessibility were developed,[15–21] most methods (except two papers on ψ angles[11,21]) on predicting backbone torsion angles are limited to discrete dihedral-angle states based on local (fragment) structural patterns.[7,12,22–27] The real-value prediction of both ϕ and ψ torsion angles was only developed recently by us.[28] Reasonable accuracy has been achieved for both solvent accessibilities and backbone torsion angles by using an integrated neural networks with a backpropagation algorithm.[21,28] In the backpropagation algorithm, the errors propagate backwards by updating neural-network weights in the direction that minimizes the error. This gradient-based algorithm, however, often leads to local minima.[29]

Many different types of methods were developed to overcome the local-minimum problem of the backpropagation algorithm. One obvious approach is to concentrate on optimization of learning rates or step sizes[30–33] and the employment of various minimization meth-

Grant sponsor: National Institutes of Health (NIH); Grant numbers: GM066049, GM068530.
*Correspondence to: Yaoqi Zhou, Indiana University School of Medicine, Walker Plaza, 719 Indiana Ave. Suite 319, Indianapolis, IN 46202. E-mail: yqzhou@iupui.edu
Received 31 March 2008; Revised 2 June 2008; Accepted 9 June 2008
Published online 14 August 2008 in Wiley InterScience (www.interscience.wiley.com).
DOI: 10.1002/prot.22193

odologies such as conjugate gradient,[34,35] Levenberg-Marguardt algorithm,[36,37] stochastic backpropagation,[38] genetic algorithm,[39,40] simulated annealing,[41,42] or a hybrid of optimization methods.[43] The second approach focuses on optimizing network architecture during training by employing genetic algorithm,[44–47] self-organized network,[48] or fuzzy logic.[49–51] The third approach develops the algorithms for estimating initial weights and uses the backpropagation algorithm for refinement. Several initialization methods such as evolutionary algorithm,[52] orthogonal least-square,[53] statistically controlled weight optimization,[54] linear least-square,[55–58] ant-colony optimization,[59] and a restricted Boltzmann machine for initial mapping,[60] were developed. Other methods developed include ensemble learning for consensus prediction,[61] boosting,[62] learning from hints[63,64] (using known information about the output to constrain learning), regularization (favoring smooth network function and avoiding over-fitting),[61] pruning (removing redundant networks),[65] and "induced learning retardation" (inhibiting the largest contributing neurons temporally).[66]

The purpose of this article is to develop improved neural network methods that are suitable for large-scale learning that requires optimization of hundreds of thousands of weights simultaneously, as in the case of predicting RSA and torsion angles. Clearly, global optimization techniques such as genetic algorithm are computationally too expensive to carry out. Here, we propose a guided weighting scheme to steer the learning to a more optimized solution. The guided weighting scheme is conceptually similar to many approaches such as learning from hints that employs known information about the output to constrain learning[63,64] and regularization that penalizes against certain models.[61] The guided weighting scheme developed here is tailored for the large-scale learning often encountered in predicting structural properties of proteins.

We have performed five experiments with different database sizes, different network architectures, and different sizes of input windows. All results reveal a consistent improvement due to guided learning. Moreover, a two-layer neural network with a bipolar activation function is effective in improving prediction accuracy, for ψ angle, in particular. All together, the resulting method reaches a new level of accuracy for predicting residue solvent accessibility and backbone torsion angles.

THEORY

Basic network architecture

Without losing generality, we consider a simple neural-network with two hidden layers. The input to the neural network will be designated by $x_{j(i)}$, where $j = 1, \ldots, j$ is an index designating the sequence position of the amino acid in a window surrounding the central residue and $i = 1, \ldots, n$ is an index for the input features of a given residue j. For the two-hidden-layer network the output values of the hidden layers will be designated by

$$h_k^1 = f(S_k^1), \text{ with } S_k^1 = \sum_{j=1}^{J} w_{jk}^1 \cdot x_j. \quad (1)$$

and

$$h_l^2 = f(S_l^2), \text{ with } S_l^2 = \sum_{k=1}^{K} w_{kl}^2 \cdot h_k^1. \quad (2)$$

where, $k = 1, \ldots, K$, $l = 1, \ldots, L$, with K, L the total number of neurons in the first and second hidden layers, respectively, $f(x)$ is the activation function, w_{jk}^1 and w_{jk}^2 are the neural network weights that connect the neurons in the input and the first hidden layer and the neurons in the first and second hidden layers, respectively. In calculating S_k^1, w_{jk}^1 is a vector of length n and the multiplication is the vector dot product.

The values of the output neurons, p_m, are obtained in a similar fashion,

$$p_m = f(S_m^3), \text{ with } S_m^3 = \sum_{l=1}^{L} w_{lm}^3 h_l^2, \quad (3)$$

where, w_{lm}^3 are the weights that connect the neurons in the second hidden layer with the neurons of the output layer, and $m = 1, \ldots, M$ with M the number of neurons in the output layer.

Training of the neural network is achieved by comparing the predicted outputs, p_m, to their known values (e.g., ψ values) for the training proteins by obtaining the sum square error E. For example, for the ψ angle

$$E(w_{jk}^1, w_{kl}^2, w_{lm}^3) = \frac{1}{2}\sum_{m=1}^{M}(\psi_m - p_m)^2. \quad (4)$$

This error function is then minimized by the steepest gradient descent method, i.e., updating the weights according to

$$\dot{w}_{jk}^1 = -\eta \frac{\delta E}{\delta w_{jk}^1}, \quad (5)$$

with η the learning rate. Similar expressions for $\dot{w}_{kl}^2$, $\dot{w}_{lm}^3$ are obtained. Note that Eq. (5) results in the minimization of the sum squared error due to the relationship $\dot{E} = \frac{\delta E}{\delta w_{jk}^1} \cdot \dot{w}_{jk}^1$. This description of the computational model is known in the neural network literature as the backpropagation method,[67] the result of Eq. (5) is to correct the weights based on the prediction error being back propagated from the output layer towards the input layer.

The above equations are for a two-hidden-layer network. Equations for the one-hidden-layer network are essentially the same. However, in this study, we will use a unipolar activation function for one-hidden-layer network [$f(x) = 1/(1 + \exp(-\alpha x))$ with $\alpha = 1$, the activation parameter that was decided upon by a process of trial and error optimization]. For the two-hidden-layer network we will use a bipolar activation function [$f(x) = \tanh(\alpha x)$, $\alpha = 0.2$ by trial and error]. We use two networks of different layers and different activation functions to test if the effect of guided learning is robust for different neural networks.

Guided neural weights

Computational neural networks can in principle approximate any continuous function, in any finite number of variables, to any degree of accuracy.[68] Stated more precisely, for any finite function $\psi(x)$ and positive number $\varepsilon > 0$ there exist a set of weights $w^i_{j\ k}$ such that the prediction of the network, $p(x)$, obeys $\|\psi(x) - p(x)\| < \varepsilon$. For the case of sequence-based structure prediction for proteins, ψ would represent the dihedral angles of the amino acids, and x would represent the amino acid sequence. Hence, the heart of the neural network is the selection of the weights. The standard approach described above is to initialize the weights in some random fashion and then use some minimization algorithm on the sum square error to train the weights. The steepest gradient descent method described above often leads a locally rather than globally optimized solution.

To go beyond the basic gradient-based backpropagation algorithm, we propose a guided learning scheme based on an intuitive pattern for neural-network weights. To do this, we treat each weight as composed of two parts,

$$w^i_{jk} = b^i_{jk} g^i_{jk}. \tag{6}$$

The first part, b^i_{jk}, is the to-be-optimized weights, whereas the second part, g^i_{jk}, is the fixed guiding factor that represents a-priori intuitive knowledge about the system (i.e. the knowledge does not have to be exact). Each of the b^i_{jk}'s is initialized to a random value in the range $[-0.5,0.5]$, whereas the g^i_{jk}'s are set at the beginning of the training and are not updated throughout the training in this study. On the other hand, if a lot of information is known about the system being predicted, such that the initial choice of the g^i_{jk} gives good predictions, a possible modification for the initial choice of the randomized weights is to set $b^i_{jk} = 1 + rnd$, where, rnd is a uniform distribution of random numbers with zero mean within some interval. In this way the b^i_{jk}'s can be used to refine the prediction given by g^i_{jk}.

As an illustrative example for this guided learning, we wish to incorporate the intuition that input features of a to-be-predicted residue should have the largest contribution to the prediction accuracy, whereas, the more distant in sequence location is a residue from the to-be-predicted residue the weaker the contribution of its input features to the prediction accuracy. This sequence-distance-dependent decay is only true for the majority of residues but not for every residue because nonlocal interactions (strong interactions between residues far from each other in sequence locations) are known to be important for stabilizing protein structures. They are yet to be included in machine learning. To implement this intuition as a guiding factor, we assume that the neural network is positioned on a two-dimensional plane and the guiding factor, g^i_{jk}, for a two-layer network are given by equations below.

$$g^1_{jk} = \frac{1}{\sqrt{1 + \left((k-1)\frac{J-1}{K-1} - (j-1)\right)^2}}, \tag{7}$$

$$g^2_{kl} = \frac{1}{\sqrt{1 + \left((k-k_c)\frac{J-1}{K-1} - (l-l_c)\frac{J-1}{L-1}\right)^2}}, \tag{8}$$

and

$$g^3_{lm} = \frac{1}{\sqrt{1 + \left((l-l_c)\frac{J-1}{L-1} - (m-m_c)\right)^2}}, \tag{9}$$

with $k_c = \frac{K+1}{2}$, $l_c = \frac{L+1}{2}$, and $m_c = \frac{M+1}{2}$, the central location of the two hidden and output layers, respectively. The guiding weights are designed so that residues that are closer (in sequence distance) to a given amino acid will contribute more in determining its corresponding structural properties. One should also note that the decay of the signals through longer connections also naturally mimics the decay in the voltage signal between far away physiological neurons. Obviously, there are many other possible equations for the guiding factors that will satisfy the same intuition. Because the purpose of this article is to validate the approach of guided learning, we did not study any other possible functional form for the guiding factors.

Given the above approach, the equations for updating the training weights are as follows. For the weights between the second hidden layer and the output layer let

$$\delta p_{lm} = \alpha(\psi_m - p_m)p_m(1 - p_m)g^3_{lm},$$

then

$$\Delta b^3_{lm} = \eta\ \delta p_{lm}\ h^2_l. \tag{10}$$

For the weights between the first and second hidden layers let

$$\delta h_{kl}^2 = \alpha h_l^2 (1 - h_l^2) g_{kl}^2 \sum_{m=1}^{M} \delta p_{lm} b_{lm}^3,$$

then

$$\Delta b_{kl}^2 = \eta \, \delta h_{kl}^2 \, h_k^1. \tag{11}$$

For the weights between the input and the first hidden layers let

$$\delta h_{jk}^1 = \alpha h_k^1 (1 - h_k^1) g_{jk}^1 \sum_{l=1}^{L} \delta h_{kl}^2 b_{kl}^2$$

then

$$\Delta b_{jk}^1 = \eta \, \delta h_{jk}^1 \, x_j. \tag{12}$$

Technical details

We conducted five experiments for testing the proposed guided learning. Experiment I uses a one-layer network, 21-residue input window, and a database of randomly selected 500 proteins from the original SPINE dataset.[69] Experiment II differs from Experiment I by employing a two-layer network. Experiment III differs from Experiment I with a larger database of 2479 proteins with length less than 500 amino acids from the original SPINE database. Experiment IV is same as Experiment III except with a two-layer network. The only difference between Experiments IV and V is that the latter uses a larger input window of 41 residues. The first two experiments of tests contain prediction of the backbone ψ angle while the other three experiments predict ψ, ϕ and residue solvent accessibility. All experiments are done twice: one with and the other without the guiding factors. Thus, we have made a total of 22 neural networks for testing the proposed method. This large number of tests is conducted to check if the performance of the guided learning depends on the database size, different properties predicted, and the size of input features. A one-layer neural network with a unipolar activation function was used in Real-SPINE for RSA prediction and Real-SPINE 2.0 for torsion angle prediction.

We use 28 input features for characterizing each residue as described in SPINE,[69] real-SPINE[21] or real-SPINE 2.0.[28] They are sequence profiles, seven representative physical parameters, and the secondary structure. The actual three-state secondary structures from DSSP[70] are used for training the weights, and predicted secondary structures from SPINE[69] are employed in testing the prediction accuracy. The terminal regions of proteins were accounted for by setting appropriate boundary conditions on the input window of the neural network. For example, with an input window of 21 residues for the first residue in the protein chain we use only residues at positions 11 to 21 in the input window. A bias is used to further refine the network. All the networks presented in this article have 101 neurons per hidden layer. In total we have $28 \times n_{window}$ features plus one bias for a given window size of n_{window} residues.

Experimental values of ψ and ϕ angles and solvent accessible surface areas are obtained from the DSSP program.[70] As introduced in a previous paper,[28] the ψ angles were shifted such that a minimum number of angles occur at the edges of the prediction window, that is, the ψ angle is shifted by adding 100° to the angles between -100° and 180°, and adding 460° to the angles between -180° and -100°. This shift ensures that a minimum number of angles occur at the ends of the sigmoidal function. This region is inherently difficult to predict in a neural-network-based machine learning method. No shift was performed for the ϕ angle since no improvement was observed in these results. Both angles are further normalized to be between $[-1,1]$ for the two-hidden-layer network (bipolar activation function) and $[0,1]$ for the one-hidden-layer network (unipolar activation function). The solvent accessibility of a residue (RSA) is obtained by its solvent accessible surface area relative to the maximum value in the dataset. Note that this is a slight departure from the method employed by Real-SPINE[21] which was based on normalizing the RSA by the accessible surface area of the residue in its "unfolded" state.[15] The reason for this departure is that the results of the DSSP program contain RSA values which are greater than the "unfolded" values given by Ahmad *et al.*[15] The normalization factors for the RSA are given in Table 2.

The reported accuracy is based on the 10-fold cross validation technique. In this procedure 90% of the data is used for training and the remaining 10% is used for testing. This process is repeated 10 times such that every protein will be part of one of the testing groups. Overfitting protection is achieved by setting aside a random 5% portion of the training data for independent testing. The training is terminated when the accuracy of the prediction does not improve for the 5% of residues set aside for 100 epochs, or when 1000 training epochs have completed. Weights corresponding to the best prediction over the 5% data are then used to give prediction for the 10% data used for validation.

We optimized learning rates by trial-and-error in testing prediction accuracy of ψ with a small dataset. We found an optimal learning rate of 0.01, which is much faster than those used in SPINE,[69] Real-SPINE[21] (0.001) and Real-SPINE 2.0 (0.0001). Thus, this learning rate is used for predicting all three properties (ψ, ϕ, and solvent accessibility). In addition, a momentum coefficient of 0.4 is used.

The quality of the prediction is evaluated by the following parameters. Mean absolute error (MAE) is the

Table I
The 10-Fold-Cross-Validated Accuracy for Predicting the ϕ and ψ Angles, and RSA from Five Experiments

	I (500,1,21)[a]		II (500,2,21)[a]		III (2479,1,21)[a]		IV (2479,2,21)[a]		V (2479,2,41)[a]	
	No[b]	Yes[c]	No	Yes	No	Yes	No	Yes	No	Yes
ψ-Q_{10}[d]	43.5%	45.3%	48.1%	49.5%	(47.0 ± 0.8)%	(48.4 ± 0.5)%	49.8 ± 0.5%	(50.7 ± 0.5)%	(48.5 ± 0.4)%	(50.1 ± 0.6)%
ψ-$Q_{10\%}$[e]	61.1%	63.0%	65.6%	66.8%	(64.6 ± 0.7)%	(65.8 ± 0.5)%	(67.3 ± 0.4)%	(68.5 ± 0.5)%	(65.7 ± 0.4)%	(67.8 ± 0.6)%
ψ-PCC[f]	0.72	0.729	0.757	0.770	0.741 ± 0.007	0.746 ± 0.007	0.743 ± 0.007	0.746 ± 0.007	0.729 ± 0.007	0.743 ± 0.007
ψ-MAE[g]	41.5°	39.8°	39.8°	38.1°	(38.3 ± 0.8)°	(37.3 ± 0.8)°	(36.8 ± 0.9)°	(36.1 ± 0.8)°	(38.2 ± 0.9)°	(36.6 ± 0.8)°
ϕ-Q_{10}[d]					(54.6 ± 0.5)%	(55.6 ± 0.5)%	(54.8 ± 0.5)%	(56.1 ± 0.5)%	(54.9 ± 0.4)%	(56.1 ± 0.4)%
ϕ-$Q_{10\%}$[e]					(81.7 ± 0.4)%	(82.1 ± 0.4)%	(82.0 ± 0.4)%	(82.4 ± 0.4)%	(81.2 ± 0.4)%	(82.2 ± 0.3)%
ϕ-PCC[f]					0.653 ± 0.005	0.658 ± 0.005	0.653 ± 0.005	0.659 ± 0.005	0.642 ± 0.006	0.654 ± 0.006
ϕ-MAE[g]					(22.8 ± 0.4)°	(22.3 ± 0.4)°	(22.6 ± 0.3)°	(22.2 ± 0.4)°	(22.8 ± 0.4)°	(22.3 ± 0.3)°
RSA-Q_{10}[d]					(39.0 ± 0.8)%	(39.7 ± 0.5)%	(39.2 ± 0.7)%	(39.9 ± 0.4)%	(38.7 ± 1.4)%	(39.7 ± 0.8)%
RSA-$Q_{10\%}$[e]					(57.0 ± 0.9)%	(58.0 ± 0.5)%	(57.4 ± 0.8)%	(58.1 ± 0.3)%	(56.5 ± 1.5)%	(57.7 ± 0.8)%
RSA-PCC[f]					0.737 ± 0.004	0.744 ± 0.005	0.738 ± 0.004	0.745 ± 0.004	0.725 ± 0.005	0.742 ± 0.004
RSA-MAE[g]					0.114 ± 0.002	0.112 ± 0.001	0.113 ± 0.002	0.111 ± 0.001	0.117 ± 0.002	0.112 ± 0.001

I, II, III, IV and V are the Experiments.
Standard Deviations between ten folds are also shown for Experiments III, IV and V.
[a]The number of proteins in the dataset, the number of hidden layers, and input window size.
[b]No guided weights.
[c]Guided weights are used.
[d]Q_{10}: Fraction of residues with correctly predicted states. Angles are divided into 10 states with 36° per bin.
[e]$Q_{10\%}$: Fraction of residues whose angles are predicted within 36° from the true value.
[f]Pearson's correlation coefficient between predicted and actual values.
[g]Mean-absolute error between predicted and actual values. Degrees are used for the ϕ and ψ angles and [0,1] normalization for the RSA.

absolute difference between predicted and actual values of a normalized structural property that are averaged over all predicted residues. Q_{10} is the fraction of residues whose angles are in correctly classified states when the torsion angles (or RSA) are equally divided into 10 states (36° per bin for angles or 0.1 per bin for RSA). We will also use $Q_{10\%}$, the fraction of residues whose predicted angles are within 36° from the actual angle value (or 10% for RSA).

The final reported result is based on a simple average of five predictors based on different random initial weights (only two for Experiment V, due to intensive computing requirement). We report Pearson's correlation coefficients for angles to compare with previous results. Other statistical tests may be more suitable for circular data such as angles.

The processing times of each epoch for weight training are 2.4 min for Experiment III, 3.2 min for Experiment IV, and 12.7 min for Experiment V on an Intel Xeon CPU model E5345 clocked at 2.33GHz. Note that these times are approximate and that guided and nonguided neural networks take approximately the same duration. Additional details regarding the method, dataset, and algorithm can be found in Ref. [28] for backbone angles and Ref. 21 for residue solvent accessibility.

RESULTS

Table I summarizes the results of five experiments that compare the prediction accuracy of the real-value ψ and ϕ angles and RSA given by the networks with and without guided learning for ψ, ϕ, and RSA. There is a consistent improvement after introducing the guided learning, regardless of the number of hidden layers, the size of input window, the size of the database for training and cross-validation, and the parameter that measures the accuracy. The absolute improvement ranges between 0.9–2.2% for Q_{10} and $Q_{10\%}$ in ψ, 0.4–1.3% for Q_{10} and Q_{10}% in ϕ, and 0.7–1.2% for Q_{10} and $Q_{10\%}$ in RSA. The mean absolute errors of ψ, ϕ and RSA are reduced by 2–4%. These improvements are often greater than the standard deviation among 10 folds. The increase of correlation coefficient due to guided learning is also observed. Positive improvements in five different experiments indicate the statistical significance of the observed improvements according to student's t-test. For example, the paired t-test[71] on five pairs of Q_{10} values (guided versus unguided) in Row 1 of Table 1 yields a P-value of 0.0007, indicating that the difference between two sets of data is statistically significant. The mean of the difference is 1.4% with 95% confidence interval of the difference from 1.0% to 1.8%.

Table I indicates that Experiment IV (2479 proteins, a two-layer network and a window size of 21 residues) yields the most accurate predictor for ψ, ϕ, and RSA. Thus, we analyze the results from Experiment IV in more detail. Table II displays the mean absolute errors for individual residues and secondary structural elements. The reduction of the error occurs essentially for every residue type and every secondary structure element for all three properties (ψ, ϕ, and RSA). The average improvement is ∼ 2%. Interestingly, residue Proline (P) has a greater improvement of 12% for the ϕ angle and 6% for the ψ angle. The only exception is that the accuracy for the

Table II
MAE for ϕ, ψ, and RSA for Residue Types and Secondary-Structure Elements Based on 10-fold Cross Validation with Experiment IV

AA	ϕ (°)		ψ (°)		RSA		SA
type	No[a]	Yes[b]	No[a]	Yes[b]	No[a]	Yes[b]	Max[c]
R	20.9	20.7	35.1	34.3	0.141	0.139	271
K	20.9	20.4	36.2	35.4	0.125	0.122	257
D	25.2	24.7	40.5	39.8	0.148	0.145	183
E	18.6	18.3	33.8	33.0	0.108	0.105	286
N	33.1	32.1	40.4	39.8	0.150	0.147	188
Q	20.0	19.7	34.1	33.4	0.145	0.141	215
H	26.7	26.4	39.9	39.3	0.126	0.124	238
Y	21.8	21.4	34.9	34.3	0.119	0.118	250
W	20.7	20.6	35.6	35.3	0.113	0.112	260
S	24.7	24.3	44.8	43.9	0.114	0.111	181
T	20.0	19.7	42.2	41.1	0.114	0.111	192
G	61.6	61.0	58.2	56.4	0.110	0.108	136
P	11.0	9.7	49.6	46.8	0.148	0.145	170
A	18.5	18.2	33.2	32.4	0.090	0.087	169
M	19.6	19.5	32.2	31.7	0.102	0.101	236
C	23.5	23.1	37.5	36.9	0.083	0.085	139
F	21.3	20.9	34.5	34.1	0.105	0.104	221
L	15.7	15.4	30.2	29.6	0.088	0.088	221
V	16.5	16.3	29.5	28.9	0.095	0.096	171
I	15.3	15.1	27.7	27.1	0.085	0.083	210
Helix	10.5	10.2	20.4	20.2	0.108	0.105	
Strand	25.7	25.3	32.5	31.6	0.089	0.087	
Coil	34.0	33.5	57.6	56.1	0.139	0.137	

[a]No guided weights.
[b]Guided weights are used.
[c]Maximum solvent accessible surface area from database (Å^2).

RSA of the Cysteine residue is reduced by 2% with the guided learning network. However, even for Cysteine there is a 2% improvement in the prediction accuracy of the ϕ and ψ angles. Similarly, 1–3% reductions of MAE for helical, strand and coil residues are observed.

We further investigate the improvement of the prediction for the different regions of the angles and RSA spaces. The Q_{10} scores for ten evenly spaced bins are given in Figure 1, 2, and 3, for the angles ϕ and ψ, and the RSA, respectively. In the figures we also give the results of a random prediction based only on the distribution of angles or RSA. Note that the random prediction bars are proportional to the frequency of occurrence of their respective bins. As before we find an improvement in the prediction accuracy of $\sim$ 2% for the most populated bins. We note that for the RSA, the Q_{10} score for the second most populated bin between 0.1 and 0.2 shows a reduction in the prediction accuracy with the introduction of guided learning. In general, guided learning makes the most improvement in the highly populated regions.

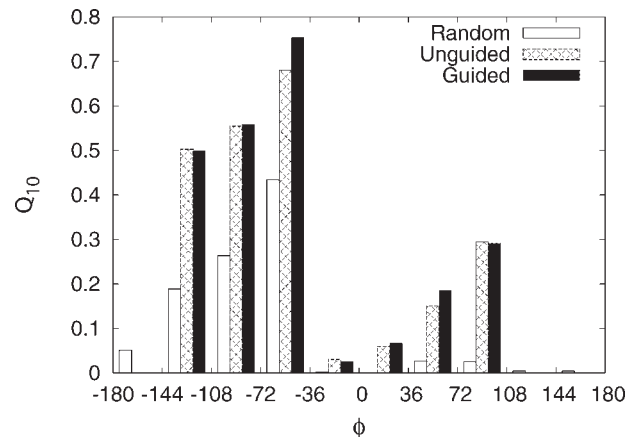

Figure 1
Q_{10} score for the ϕ angle for 10 evenly spaced bins.

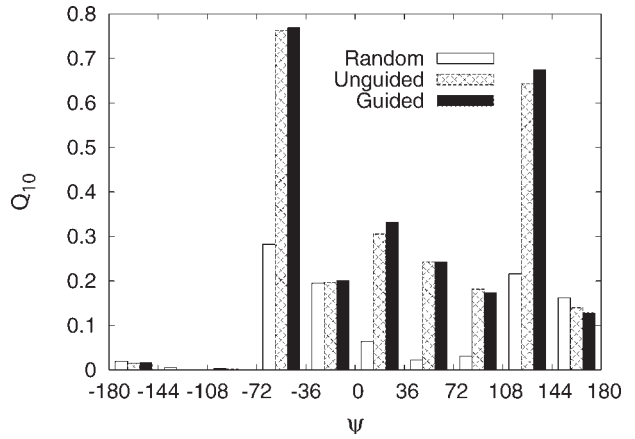

Figure 2
Q_{10} score for the ψ angle for 10 evenly spaced bins.

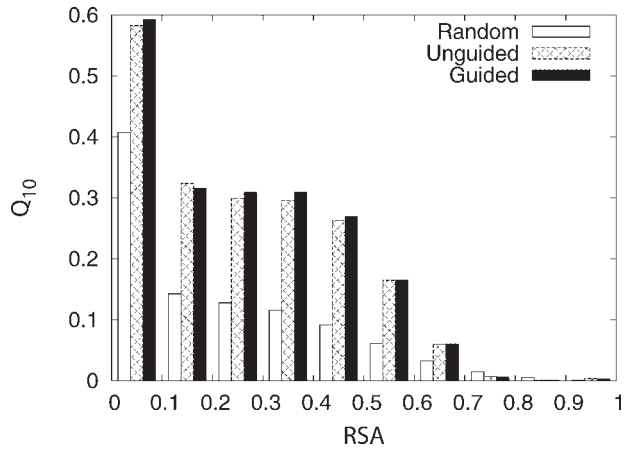

Figure 3
Q_{10} score for the residue surface accessibility for 10 evenly spaced bins with a [0,1] normalization.

Table III
The Effect of Long Chains and Learning Rates on the 10-Fold-Cross-Validated Accuracy for Predicting the ϕ and ψ Angles, and RSA

Method	(2479,0.01)[a]	(2640,0.01)[b]	(2640,0.001)[c]
ψ-Q_{10}[d]	(50.7 ± 0.5)%	(49.7 ± 0.4)%	(49.7 ± 0.5)%
ψ-$Q_{10\%}$[e]	(68.5 ± 0.5)%	(67.5 ± 0.4)%	(67.5 ± 0.5)%
ψ-PCC[f]	0.746 ± 0.007	0.743 ± 0.006	0.743 ± 0.006
ψ-MAE[g]	36.1 ± 0.8°	(36.4 ± 0.7)°	(36.3 ± 0.7)°
ϕ-Q_{10}[d]	(56.1 ± 0.5)%	(56.1 ± 0.6)%	(56.0 ± 0.5)%
ϕ-$Q_{10\%}$[e]	(82.4 ± 0.4)%	(82.1 ± 0.3)%	(81.2 ± 0.3)%
ϕ-PCC[f]	0.659 ± 0.005	0.656 ± 0.005	0.656 ± 0.005
ϕ-MAE[g]	22.2 ± 0.4°	(22.1 ± 0.3)°	(22.2 ± 0.2)°
RSA-Q_{10}[d]	(39.9 ± 0.4)%	(40.2 ± 0.5)%	(40.1 ± 0.6)%
RSA-$Q_{10\%}$[e]	(58.1 ± 0.3)%	(58.2 ± 0.4)%	(58.0 ± 0.5)%
RSA-PCC[f]	0.745 ± 0.004	0.739 ± 0.005	0.739 ± 0.006
RSA-MAE[g]	0.111 ± 0.001	0.111 ± 0.001	0.111 ± 0.001

[a]Experiment IV for the dataset of 2479 proteins.
[b]Experiment IV for the dataset of 2640 proteins (including chains with more than 500 residues).
[c]Experiment IV for the dataset of 2640 proteins but with a slow learning rate of 0.001.
[d]Q_{10}: Fraction of residues with correctly predicted states. Angles are divided into 10 states with 36° per bin. Residue solvent accessibility (RSA) is divided into 10 states with 0.1 per bin.
[e]$Q_{10\%}$: Fraction of residues whose angles are predicted within 36° (or 10% for RSA) from the true value.
[f]Pearson's correlation coefficient between predicted and actual values.
[g]Mean-absolute error between predicted and actual values. Degrees are used for the ϕ and ψ angles and [0,1] normalization for the RSA.

In addition to guided learning, changing from one-layer with a unipolar activation function to a two-layer neural network with a bipolar activation function also leads to significant improvement. The effect is the largest for ψ. For example, there is a 2.3% absolute improvement from 48.4% to 50.7% in Q_{10} with a database of 2479 proteins. However, the corresponding improvements are only 0.5% (from 55.6% to 56.1%) for ϕ and 0.2% for solvent accessibility. Thus, changing neural network architecture affects different structural properties differently. Both guided learning and changing the neural network architecture improve the prediction accuracy.

To facilitate the comparison with previous work, we further performed Experiment IV on the original data set of 2640 proteins[69,21,28] that includes 161 proteins with more than 500 residues. Results are shown in Table III. The accuracy based on the 10-fold-cross validated values from the database of 2640 proteins is essentially the same as that from the database of 2479 proteins for ϕ and residue solvent accessibility but is slightly worse for ψ (<2% in relative difference). Note that the guided learning for the 2640 protein dataset also improves the prediction accuracies by ∼ 2% relative to unguided neural networks (results not shown). We also tested the effect of learning rates on the accuracy. No significant effect is observed (Table III).

It is also of interest to know if a method trained with short chains is useful to predict structural properties of long chains. We apply the method trained with the database of 2479 proteins (chain length <500) to 161 proteins with chain length of more than 500 amino acid residues. We obtained 64.8%, 81.0%, 55.2% for $Q_{10\%}$ of ψ, ϕ, and RSA, respectively. This was done from the average of 10 sets of the results generated from weight parameters from 10 fold training with the database of 2479 proteins. The corresponding $Q_{10\%}$ accuracies (10-fold-cross-validated) are 64.8%, 81.1%, and 57.1%, respectively, when long chain proteins are used as a part of training and 10-fold—cross-validation. Thus, only the accuracy of RSA is improved when long chain proteins are included in training and test sets.

DISCUSSION

We have introduced a machine learning technique called guided learning. The purpose of guided learning is to guide the neural network based on prior knowledge or intuition on neural network weights. The idea is tested using five different experiments and three structural properties of proteins. A consistent 2% reduction in mean absolute error is observed regardless the size of database, the number of hidden layers, the size of input window, the residue type or secondary structure type. Thus, the observed improvement is robust and statistically significant. Such an improvement is obtained without any significant increase in computational time. This is important because we are optimizing a large number of weights simultaneously ((28 × 21 + 1) × 101 + 102 × 101 + 102 weights for experiment IV).

Although a 2% improvement may seem small, it is significant for protein structural properties. For example, the accuracy of secondary structure prediction has been stagnated around 77%[13] until the 10-fold—cross-validated 80% that was reached recently.[69] In a separate study, we found that this technique leads to 1% improvement in secondary structure prediction (Q_3 = 81%, Faraggi and Zhou, in preparation). Moreover, this study represents only a preliminary implementation of the proposed guided learning technique to a few specific cases. Application to other problems with better defined "intuitions" may be more profitable. Furthermore, the functional form for guiding factors [Eqs. (7–9)] used in this study may not be optimal. Another possibility to improve guided learning is to develop an iterative method to improve the guiding factors.

Introducing the sequence-distance-dependent decay as a guiding factor also makes physical sense. It mimics the natural processes associated with natural neural networks. Due to the resistance of the axon connecting different natural neurons there will be a potential drop as a signal is propagated from one neuron to the next. Since one neuron may be connected to several others, with axons of different lengths connecting them, the outcome will be that neurons that are connected by longer axons will propagate weaker signals between them. This is exactly the effect of the guiding factors introduced here.

Table IV
The 10-Fold-Cross-Validated Accuracy for Predicting the ϕ and ψ angles, and RSA

	ψ		ϕ		RSA	
Criterion	Real-SPINE[a]	This work[b]	Real-SPINE[a]	This work[b]	Real-SPINE[c]	This work[b]
Q_{10}[d]	46.6%	49.7%	45.0%	56.1%	34.7%	40.2%
$Q_{10\%}$[e]	64.7%	67.5%	80.1%	82.1%	—	58.2%
PCC[f]	0.745	0.743	0.707	0.656	0.738	0.739
MAE[g]	38.2°	36.4°	24.8°	22.1°	0.142	0.111

The Comparison Between Prediction Accuracies of This Study and Best Reported Accuracies.
[a]Real-SPINE 2.0[28] for the dataset of 2640 proteins.
[b]Experiment IV for the dataset of 2640 proteins.
[c]Real-SPINE[21] for the dataset of 2640 proteins.
[d]Q_{10}: Fraction of residues with correctly predicted states. Angles are divided into 10 states with 36° per bin. Residue solvent accessibility (RSA) is divided into 10 states with 0.1 per bin.
[e]$Q_{10\%}$: Fraction of residues whose angles are predicted within 36° (or 10% for RSA) from the true value.
[f]Pearson's correlation coefficient between predicted and actual values.
[g]Mean-absolute error between predicted and actual values. Degrees are used for the ϕ and ψ angles and [0,1] normalization for the RSA.

It is of interest to compare the prediction accuracy to the best reported accuracy from Real-SPINE[21] for solvent accessibility and Real-SPINE 2.0[28] for torsion angles. Table IV shows that there are 3%, 11%, and 5.5% absolute increases in Q_{10} score of ψ, ϕ, and RSA, respectively, 3% and 2% absolute increases in $Q_{10\%}$ score of ψ and ϕ, respectively, and 5%, 10%, and 22% reduction of MAE of ψ, ϕ, and RSA, respectively. Interestingly, we found that there is a reduction, rather than an increase, of correlation coefficient from 0.707 to 0.656 for ϕ. We found that this is mainly due to the fact that the correlation coefficients were calculated between shifted ϕ angles in Real-SPINE and unshifted angles in this work. If shifted angles are converted back to normal values, the correlation coefficient will reduce from 0.707 to 0.53 in Real-SPINE2.0. This result highlights the fact that correlation coefficients are unsuitable for circular data such as angles. Moreover, correlation coefficient ignores the possible complex distribution pattern of the variables to be correlated.[72] Both ϕ and ψ angles have a bimodal rather than a normal distribution.

The overall improvement on ψ angles can be mostly interpreted with improvement due to introduction of guided learning and a two-layer neural network. The more significant improvements for ϕ and RSA exist prior to the introduction of the two-layered network (Experiment III in Table I). For both the RSA and the ϕ angle we have found that the new scaling introduced here proves beneficial. We also found that the faster learning rate of 0.01 yields no improvement but allows a much faster convergence of the neural network training.

The predicted angles and residue surface accessibility will likely be useful for improving fold recognition and conformational sampling of protein structures. This was demonstrated in a number of earlier studies for surface accessibility.[1–6] The predicted angles have been also used to improve fold recognition,[7–9] sequence alignment,[10] and the accuracy of secondary structure prediction.[11,12] The work presented here not only provides a new technique for machine learning but also an improved prediction tool available on http://sparks.informatics.iupui.edu, which is useful for protein structure prediction.

REFERENCES

1. Cheng J, Baldi P. A machine learning information retrieval approach to protein fold recognition. Bioinformatics 2006;22:1456–1463.
2. Rost B.TOPITS: threading one-dimensional predictions into three-dimensional structures. Third International Conference on Intelligent Systems for Molecular Biology. AAAI Press; 1995; pp 314–321.
3. Rost B, Sander C. Protein fold recognition by prediction-based threading. J Mol Biol 1997;270:471–480.
4. Przybylski D, Rost B. Improving fold recognition without folds. J Mol Biol 2004;341:255–269.
5. Qiu J, Elber R. SSALN: an alignment algorithm using structure-dependent substitution matrices and gap penalties learned from structurally aligned protein pairs. Proteins 2006;62:881–891.
6. Liu S, Zhang C, Liang S, Zhou Y. Fold recognition by concurrent use of solvent accessibility and residue depth. Proteins 2007;68:636–645.
7. Karchin R, Cline M, Mandel-Gutfreund Y, Karplus K. Hidden Markov models that use predicted local structure for fold recognition: alphabets of backbone geometry. Proteins 2003;51:504–514.
8. Wu S, Zhang Y. MUSTER: improving protein sequence profile-profile alignments by using multiple sources of structure information. Proteins 2008;72:547–556.
9. Zhang W, Liu S, Zhou Y. SP[5]: improving protein fold recognition by using predicted torsion angles and profile-based gap penalty. PLoS ONE 2008;3:e2325.
10. Huang YM, Bystroff C. Improved pairwise alignments of proteins in the twilight zone using local structure predictions. Bioinformatics 2006;22:413–422.
11. Wood MJ, Hirst JD. Protein secondary structure prediction with dihedral angles. Proteins 2005;59:476–481.
12. Mooney C, Vullo A, Pollastri G. Protein structural motif prediction in multidimensional phi-psi space leads to improved secondary structure prediction. J Comput Biol 2006;13:1489–1502.
13. Rost B. Review: protein secondary structure prediction continues to rise. J Struct Biol 2001;134:204–218.

14. Offmann B, Tyagi M, deBrevern AG. Local protein structures. Curr Bioinfo 2007;2:165–202.
15. Ahmad S, Gromiha MM, Sarai A. Real value prediction of solvent accessibility from amino acid sequence. Proteins 2003;50:629–635.
16. Yuan Z, Huang B. Prediction of protein accessible surface areas by support vector regression. Proteins 2004;57:558–564.
17. Adamczak R, Porollo A, Meller J. Accurate prediction of solvent accessibility using neural networks-based regression. Proteins 2004; 56:753–767.
18. Garg A, Kaur H, Raghava G. Real value prediction of solvent accessibility in proteins using multiple sequence alignment and secondary structure. Proteins 2005;61:318–324.
19. Wang J, Lee H, Ahmad S. Prediction and evolutionary information analysis of protein solvent accessibility using multiple linear regression. Proteins 2005;61:481–491.
20. Xu Z, Zhang C, Liu S, Zhou Y. QBES: predicting real values of solvent accessibility from sequences by efficient, constrained energy optimization. Proteins 2006;63:961–966.
21. Dor O, Zhou Y. Real-SPINE: an integrated system of neural networks for real-value prediction of protein structural properties. Proteins 2007;68:76–81.
22. Kang HS, Kurochkina NA, Lee B. Estimation and use of protein backbone angle probabilities. J Mol Biol 1993;229:448–460.
23. Bystroff C, Thorsson V, Baker D. HMMSTR: a hidden Markov model for local sequence-structure correlations in proteins. J Mol Biol 2000;301:173–190.
24. deBrevern AG, Etchebest C, Hazout S. Bayesian probabilistic approach for predicting backbone structures in terms of protein blocks. Proteins 2000;41:271–287.
25. deBrevern AG, Benros C, Gautier R, Valadie H, Hazout S, Etchebest C. Local backbone structure prediction of proteins. In Silico Biol 2004;4:381–386.
26. Kuang R, Lesliei CS, Yang A-S. Protein backbone angle prediction with machine learning approaches. Bioinformatics 2004;20:1612–1621.
27. Zimmermann O, Hansmann UHE. Support vector machines for prediction of dihedral angle regions. Bioinformatics 2006;22:3009–3015.
28. Xue B, Dor O, Faraggi E, Zhou Y. Real value prediction of backbone torsion angles. Proteins 2008;72:427–433.
29. Sutton R. Two problems with back-propagation and other steepest-descent learning procedures for networks. Hillsdale, NJ Erlbaum; 1986.
30. Jacobs RA. Increased rates of convergence through learning rate adaptation. Neural Netw 1988;1:295–307.
31. Wilson DR, Martinez TR. The general inefficiency of batch training for gradient descent learning. Neural Netw 2003;16:1429–1451.
32. Inazawa H, Cottrell GW. Phase space learning in an autonomous dynamical neural network. Neurocomputing 2006;69:2340–2345.
33. Wang C-H, Kao C-H, Lee W-H. A new interactive model for improving the learning performance of back propagation neural network. Autom Construct 2007;16:745–758.
34. Barnard E. Optimization for training neural nets. IEEE Transactions on Neural Networks 1992;3:232–240.
35. Charalambous C. Conjugate gradient algorithm for efficient training of artificial neural networks. Circuits, Devices and Systems, IEE Proceedings G 1992;139:301–310.
36. Hagan M, Menhaj M. Training feedforward networks with the Marquardt algorithm. IEEE Trans Neural Netw 1994;5:989–993.
37. Karelson M, Dobchev D, Kulshyn O, Katritzky A. Neural networks convergence using physicochemical data. J Chem Inf Model 2006;46:1891–1897.
38. LeCun Y, Bottou L, Orr GB, Mueller KR. Efficient BackProp. Lect Notes Comput Sci 1998;1524:5–50.
39. Janson DJ, Frenzel JF. Training product unit neural networks with genetic algorithms. IEEE Expert 1993;8:26–23.
40. Bengio S, Bengio Y, Cloutier J. Use of genetic programming for the search of a new learning rule for neural networks. In International Conference on Evolutionary Computation; Orlando: IEEE 1994;324–327.
41. Boese KD, Kahng AB. Simulated annealing of neural networks: The 'cooling' strategy reconsidered. In Proceedings of IEEE International Symposium on Circuits and Systems; Chicago, 1993;2572–2575.
42. Porto V, Fogel D, Fogel L. Alternative neural network training methods. IEEE Expert 1995;10:16–22.
43. Zanchettin C, Ludermir TB. Hybrid technique for artificial neural network architecture and weight optimization. In: Jorge A, Torgo L, Brazdil P, Camacho R, Gama J, editors. Knowledge discovery in databases: PKDD 2005, Vol. 3721. Porto, Portugal: Springer; 2005;709–716.
44. Koza JR, Rice JP. Genetic generation of both the weights and architecture for a neural network. In International Joint Conference on Neural Networks, IJCNN-91; Vol. II. Washington State Convention and Trade Center, Seattle, WA, USA: IEEE Computer Society Press; 1991.
45. Leung FHF, Lam HK, Ling SH, Tam PKS. Tuning of the structure and parameters of a neural network using an improved genetic algorithm. IEEE Trans Neural Netw 2003;14:79–88.
46. Tsai J-T, Chou J-H, Liu T-K. Tuning the structure and parameters of a neural network by using hybrid Taguchi-genetic algorithm. IEEE Trans Neural Netw 2006;17:69–80.
47. Rivero D, Dorado J, Rabunal JR, Pazos A, Pereira J. Artificial neural network development by means of genetic programming with graph codification. Proc World Acad Sci Eng Tech 2006;15:209–214.
48. Widyanto MR, Nobuhara H, Kawamoto K, Hirota K, Kusumoputro B. Improving recognition and generalization capability of back-propagation nn using a self-organized network inspired by immune algorithm (SONIA). Appl Soft Comput 2005;6:72–84.
49. Jang J-SR. Self-learning fuzzy controllers based on temporal back propagation. IEEE Trans Neural Netw 1992;3:714–723.
50. Lin C-T, Lee C. Reinforcement structure/parameter learning for neural-network-based fuzzy logic control systems. IEEE Trans Fuzzy Systems 1994;2:46–63.
51. Huang X-M, Yi J-K, Zhang Y-H. A method of constructing fuzzy neural network based on rough set theory. Proceedings of 2nd international conference on machine Learning and cybernetics 2003;3:1723–1728.
52. Hüsken M, Goerick C. Fast learning for problem classes using knowledge based network initialization. In INNS-ENNS International Joint Conference on Neural Networks (IJCNN 2000), Vol. 6; Como, Italy, 2000;619–624.
53. Colla V, Reyneri L, Sgarbi M. Orthogonal least square algorithm applied to the initialization of multi-layer perceptrons. In Proceedings of 7th European Symposium on Artificial Neural Networks; Bruges, Belgium, 1999;363–369.
54. Drago GP, Ridella S. Statistically controlled activation weight initialization (SCAWI). IEEE Transactions on Neural Networks 1992;3:899–905.
55. Biegler-König F, Bärmann FA. Learning algorithm for multilayered neural networks based on linear least squares problems. Neural Netw 1993;6:127–131.
56. Cho SY, Chow TWS. Training multilayer neural networks using fast global learning algorithm: least squares and penalized optimization methods. Neurocomput 1999;25:115–131.
57. Castillo E, Fontenla-Romero O, Bertha Guijarro-Berdi N, Alonso-Betanzos AA. Global optimum approach for one-layer neural networks. Neural Comput 2002;14:1429–1449.
58. Erdogmus D, Fontenla-Romero O, Principe J, Alonso-Betanzos A, Castillo E. Linear-least-squares initialization of multilayer perceptrons through backpropagation of the desired response. IEEE Trans Neural Netw 2005;16:325–337.
59. Liu Y-P, Wu M-G, Qian J-X. Evolving neural networks using the hybrid of ant colony optimization and bp algorithms advances in neural networks. In: Wang J, Yi Z, Zurada JM, Lu B-L, Yin H, editors. Proceedings of Third International Symposium on Neural Networks Vol. 3971. Lect Notes Comput Sci Springer; Chengdu, China, 2006;714–722.

60. Hinton GE, Salakhutdinov RR. Reducing the dimensionality of data with neural networks. Science 2006;313:504–507.
61. Bishop CM. Neural Networks for pattern recognition. Oxford University Press; 1995.
62. Freund Y, Schapire RE. A decision-theoretic generalization of on-line learning and an application to boosting. J Compu Syst Sci 1997;55:119–139.
63. Abu-Mostafa YS. Hints and the vc dimension. Neural Comput 1993;5:278–288.
64. Mrázová I, Wang D. Improved generalization of neural classifiers with enforced internal representation. Neurocomput 2007;70:2940–2952.
65. Sietsma J, Dow RJF. Creating artificial neural networks that generalize. Neural Netw 1991;4:67–79.
66. Bandibas JC, Kohyama K. An efficient artificial neural network training method through induced learning retardation: inhibited brain learning. In Proceedings Asian Conference on Remote Sensing; Taipei, Taiwan, 2006;1–3.
67. Rumelhart DE, Hinton GE, Williams RJ. Learning representations by back-propagating errors. Nature 1986;323:533–536.
68. Looney CG. Pattern recognition using neural networks. New York: Oxford University Press; 1997.
69. Dor O, Zhou Y. Achieving 80% ten-fold cross-validated accuracy for secondary structure prediction by large-scale training. Proteins 2007;66:838–845.
70. Kabsch W, Sander C. Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features. Biopolymers 1983;22:2577–2637.
71. GraphPad Software. Available at www.graphpad.com/quickcalcs/ttest1.cfm.
72. Press W, Teukolsky S, Vetterling W, Flannery B. Numerical recipes in C. 2nd ed. Cambridge, UK: Cambridge University Press; 1992.