A Unified Framework for Using Neural Networks To Build QSARs

Ajay

Department of Pharmaceutical Chemistry, University of California San Francisco, Box 0446, S-926, San Francisco, California 94143

Received April 29, 1993•

We propose a new neural network architecture that explicitly separates linear and nonlinear contributions to the biological activity. To facilitate the use of neural networks as a regular tool we demonstrate that (1) a perceptron with linear output units is equivalent to multiple linear regression and (2) one hidden unit at a time can be added to the network so that QSAR data can be modeled by everything from the simplest linear hypersurfaces to complicated ones. The significant improvements accrued by the use of weight decay are demonstrated. We conclude that models built without attempting weight decay may not be reliable either for interpretation or extrapolation. Finally we compare models generated by neural networks, rank regression, and standard regression on non-normally distributed data and conclude that neural networks like rank regression bring out many facets of the data that are inaccessible to multiple linear regression. All the experiments were done on either triazine inhibition of pure DHFR from L1210 leukemia cells and on the inhibition of intact L1210 leukemia cells sensitive and resistant to methotrexate or on steroid binding to progesterone.

Introduction

Quantitative structure–activity relationships (QSARs) have mostly been studied using multiple linear regression. A $\log(\beta 10^{\pi} + 1)$ term parameterizing a roughly parabolic dependence on π is also often invoked. Recently neural networks have successfully been used in building QSARs.[1,2] The main reason for this is that QSAR surfaces often have many kinks and wrinkles that cannot be modeled by linear hypersurfaces. However, most applications of neural networks to QSAR have neglected certain key issues like appropriate network architecture, learning algorithm and its convergence, data distribution, and the necessity of adding a regularization (weight decay) term to the error function. In this paper we demonstrate how attention to these factors leads to simpler and statistically sound models.

QSAR work using neural networks has primarily used nonlinear functions (units) to represent the input–output mapping. Using only nonlinear units makes interpretations of the model hard. It is possible to construct neural networks that intelligently mix linear and nonlinear units so that we can model everything from linear hypersurfaces to surfaces with many undulations in a straightforward way. We propose such a network so that linear and nonlinear contributions to the biological activity are easily separated. It also helps in providing a unified framework for using neural networks to build QSARs. This is possible because we can start with a linear model (perceptron with linear output units), and if that is not appropriate we can add nonlinearities to the model—one hidden unit at a time. This introduces neural networks as a tunably nonlinear method to perform regressions.

Neural network models are nonparametric. Hence it is easy to build nonrobust models. Robust models exhibit good generalization and lower sensitivity to noise. We demonstrate that incorporating a regularization term, "weight decay", is crucial to building robust models. In addition it leads to faster and better convergence.

A third point that is addressed is the influence of non-normally distributed data on neural network models. Pleiss[4] worked on such data and used rank regression (a nonparametric regression technique). This resulted in a much better model compared to the one obtained from multiple linear regression. We compare rank regression to neural network regression in the last part of this paper.

Before neural networks can be used as a regular tool, attention must be paid to other questions such as the influence of many outliers in the data and the relevance of preclustering the data. These issues are explored elsewhere.[3]

Data Sets

As it is our intention to study methodological problems for building neural network regressors, we concentrate on two well-studied data sets. The first is triazine inhibition of pure DHFR (dihydrofolate reductase) from L1210 leukemia cells as well as the inhibition of intact L1210 cells, both sensitive and resistant to methotrexate (MTX).[5] The second example is from an older work of steroid binding to pregesterone receptors.[6]

Triazine. In what follows, we state the salient features of the work by Selassie *et al.*

3-X-triazine Inhibition of Purified DHFR from L1210 Leukemia Cells Resistant to Methotrexate. The QSAR developed for this set of compounds uses $\{\pi', \log(\beta 10^{\pi'} + 1), \sigma\}$ as independent variables. π' indicates that for substituents of type $-CH_2ZC_6H_4Y$ and $-ZCH_2C_6H_4Y$ (Z = O, S, Se), $\pi_Y = 0$. Also, for all alkoxy groups (methoxy to tetradecyloxy), $\pi' = 0$. σ is the Hammet constant. They used 58 out of the 61 (the other three were dropped from the regression because they do not fit well) compounds, and the model yielded a correlation coefficient $\rho = 0.9$, and a standard deviation $s = 0.26$.

3-X-triazine Inhibition of Cultured L1210 Leukemia Cells Sensitive to Methotrexate. The QSAR developed for this set of compounds (including 61 of the 64 componds) involved $\{\pi \log(\beta 10^{\pi} + 1), \sigma, I_R, I_{OR}\}$ yielding a ρ of 0.89 and an s of 0.24. I_R is the indicator variable that takes on a value of one for alkyl groups, and I_{OR} is one for alkoxy groups. Note that π and not π' gives better results.

• Abstract published in *Advance ACS Abstracts*, October 1, 1993.

0022-2623/93/1836-3565$04.00/0

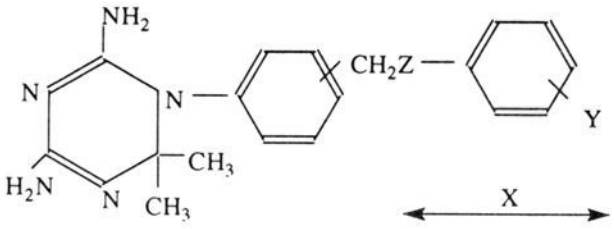

Figure 1. Structure of the triazines used in this work.

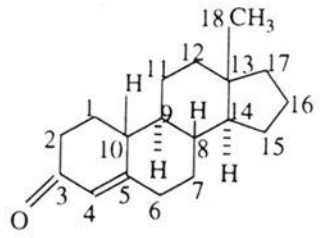

Figure 2. The parent ring system for the steroid data.

3-X-triazine Inhibition of Cultured L1210 Leukemia Cells Resistant to Methotrexate. The QSAR developed for this set of compounds (including 62 of the 64 compounds) used $\{\pi, MR\}$, yielding a ρ of 0.94 and an s of 0.22. MR represents molar refractivity. This is a very different QSAR compared to the previous ones as it is linear in all the physicochemical parameters. Selassie *et al.*, for reasons that are not clear, guessed that the optimum value of π was about 6. The optimum π for the previous cases, however (this can be derived mathematically from the roughly parabolic dependence), is about 1.8. They conclude that the 4.2 log units difference implies that more lipophilic drugs are needed for methotrexate-resistant tumors.

The structure of the triazines used in this study is illustrated in Figure 1. For a list of the compounds used and their physicochemical parameters, please refer either to the original work or to the supplementary material.

Steroid. Pleiss[4] reinvestigated the QSAR using rank regression on data originally studied by Lee *et al.*,[6] who had used linear regression. The QSAR was built for 55 androst-4-en-3-one derivatives. Pleiss reexamined the data because on extrapolation to a test set of 10 new compounds, Lee *et al.* obtained errors, $[Y_{obs} - Y_{cal}]/Y_{obs}$, ranging from 5% to 141%. The original equation of Lee *et al.* (eq 11 in ref 6) uses $\{\pi_a, \pi_b, SAI, SAO, MK, CC\}$. Here π_b is the π values of all polar groups in the 17α, 20α, and 20β substituents, and π_a is the π value of all polar groups in other positions. SAI is the surface area in hydrophobic pockets; SAO is the surface area out of hydrophobic pockets. SAI gives the net change in surface area in positions 6α, 11β, 16α, 17α, and 17β for progesterone derivatives, position 16 for androstance derivatives, and position 21 in data obtained from rabbits. SAO gives the net change in surface area in all the other positions. MK is an indicator variable indicating the presence or absence of ketones. CC is also an "indicator variable".[10] It is the sum of the number of carbon atoms with changes in hybridization from a 4-androsten-3-one parent molecule. For details on why these physicochemical parameters were chosen, the reader is referred to Lee *et al.*[6] They used 55 compounds and obtained $\rho = 0.88$ and $s = 0.54$.

The parent molecule is shown in Figure 2. The compounds forming the data set and the physicochemical parameters can be obtained either from the original reference or through the supplementary material.

Methods

Most published QSAR studies fall into two distinct but related categories. Some, like Selassie *et al.*, are interested in determining the physicochemical parameters that affect biological activity.

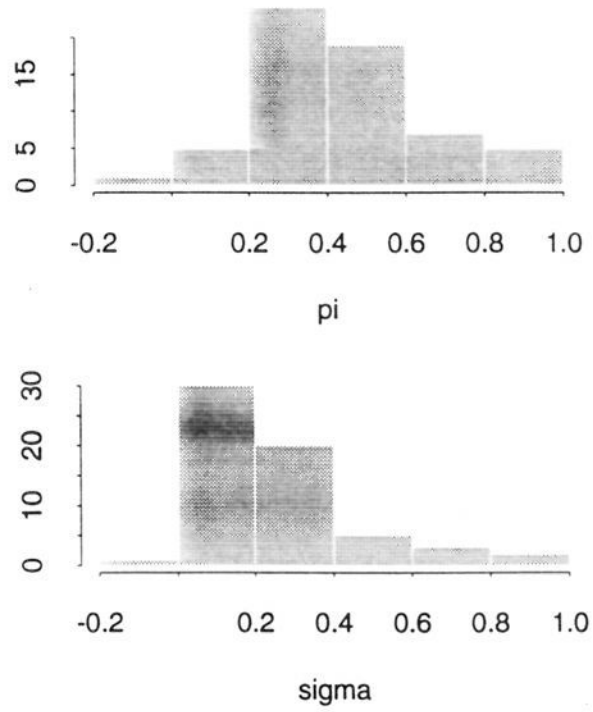

Figure 3. The distribution of π and σ as histograms. Note that π is roughly normal whereas σ is not. A similar conclusion can be drawn from a comparison of the boxplot, density plot and qqplot (not shown).

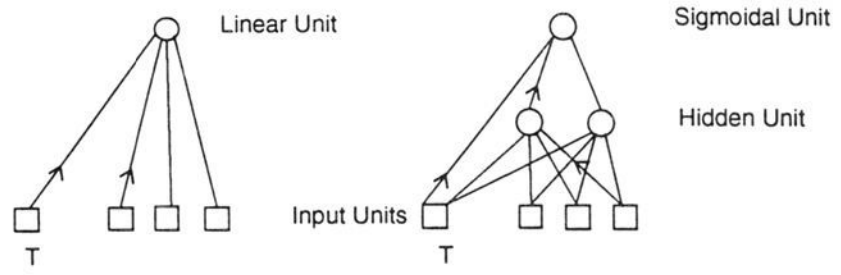

Figure 4. The architecture of a perceptron (left) and the standard neural network architecture (right) used in all the applications of neural networks to QSAR so far. "T" is the threshold unit.

They concentrate on biological and chemical questions, ignoring to a certain extent statistical soundness. There is some justification for this because of small size of the data sets. Others, like Pleiss, would like to extract as much information as possible from the data set so that generalization ability is enhanced. In this paper we would like to demonstrate the versatility and utility of our techniques to workers in both camps. Therefore, we reexamine the data sets, closely following the goals of Selassie *et al.*[5] and Pleiss.[4] Our primary interest is in bringing out salient features of neural network modeling.

We began the study by exploring the distribution of the data using standard graphical techniques. The S-PLUS package was used for this purpose. We analyzed the histogram, boxplot, density plot, and normal qqplot of π, σ, and other variables to check for normality.

Neural Network. In this section a brief description of neural networks is provided. For a detailed discussion see, for example, Hertz *et al.*[7]

Figure 4 shows the connectivity pattern for a perceptron and the standard network used in previous studies of neural networks applied to QSAR. In the standard architecture all the inputs are connected to a layer of hidden units which in turn are connected to the output. The information is passed from the input units to the output unit using

$$O_i^{\mu} = f\left(\sum_j W_{ij} f\left(\sum_k w_{jk} \xi_k^{\mu}\right)\right) \qquad (1)$$

where O_i^{μ} denotes the output of unit i for pattern μ. Note that i represents the output unit, j the hidden units, and k the input units. W_{ij} and w_{jk} are connections from the hidden to output, and input to hidden units, respectively. ξ_k^{μ} labels the kth input of pattern μ, and the function $f(\cdot)$ is a squashing function characterizing the type of the unit used in the network. We use a sigmoid unit where $f(h) = 1/[1 + \exp(-h)]$. Since the range of the sigmoid function is between zero and one, both the input and the output have to be scaled. This is usually achieved by using

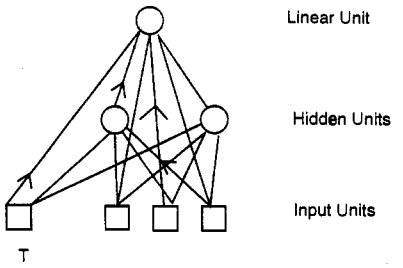

Figure 5. The architecture used in this work. The output units are linear, and there are direct connections from the input to the output.

$$x_{\text{scaled}} = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{2}$$

where $x_{\min}$ and $x_{\max}$ are the minimum and maximum values of variable x available in the data set. (It is not essential to scale the inputs or to use this particular one, but it usually works well.)

In this paper, we concentrate on a different network architecture. This is shown in Figure 5. Here, the output is determined using the relation

$$O_i^{\mu} = \sum_k W_{ik}\xi_k^{\mu} + \sum_j W_{ij} f\left(\sum_k w_{jk}\xi_k^{\mu}\right) \tag{3}$$

Since the output unit is *linear* the weights W_{ik} account for the linear relationship between the input and the output. The second term in eq 3 accounts for the deviation from linearity. Such a separation is particularly useful because of the success of linear regression in QSAR. The best way to see the decrease in complexity of the model using the new architecture is to write out eq 3 and eq 1, substituting the sigmoidal functions for $f(\cdot)$. Another advantage of linear outputs is that scaling of biological activity is no longer necessary. Therefore, only the inputs are scaled.

Training and Weight Decay. The standard backpropagation algorithm used in training such feed-forward networks is a straightforward gradient descent procedure. Under certain circumstances a simple gradient descent is inefficient. One such situation is when the number of data points is small as is usually the case in QSAR work. We, therefore, use a *quasi-Newton* method to determine the weights that minimize the squared error between the predicted and observed biological activity. We use the BFGS algorithm.[8] This procedure also avoids the problem of choosing the usual learning rate and momentum parameters used in the standard backpropagation training algorithm. We also calculate the Hessian of the error function to see if the procedure has converged to a local minimum or whether it is in some shallow basin. Training is continued until the error decrease after each pass through the data is small, and the Hessian suggests that we are in a local minimum. This enables us to detect and avoid spurious convergence.

Weight decay is a simple technique that de-emphasizes large weights. It can be viewed as a method that builds in an apriori bias toward simple models. For large weight vectors the activation of its corresponding unit would be close to the extremes of the sigmoidal function. During training this may often turn out to be the "incorrect" extreme, and it takes a long time to change them to the opposite extreme. This saturation of the hidden units may also lead to the network getting stuck in flat regions without reaching a local minimum. "Weight decay", as the term implies, decreases the magnitude of all the weights in the network at each iteration. Therefore, networks trained with weight decay often achieve lower training error, need fewer training epochs, and yield better models for generalization by decreasing the magnitude of unimportant weights. Generalization is improved because it forces the network to discover regularities in the training set instead of simply using a look-up table. Another advantage is that it makes the parameters of the network (the weights) dependent on one another. In contrast to learning without weight decay, it helps us to consider regressors with a larger ratio of weights to data points without over-fitting.

We therefore incorporate a weight decay term into the error function using

$$\frac{\partial E}{\partial w_{ij}} := \frac{\partial E}{\partial w_{ij}} + 2(\text{decay})w_{ij} \tag{4}$$

Each time the gradient is calculated, we add a penalty term to the error E (see Hertz *et al.*[7]). The ":=" symbol assigns the right hand side to the left. Note that the change in weight is proportional to the negative of the gradient calculated in eq 4. Therefore, the decay term decreases the magnitude of the weights at each step. The decay parameter should *not* be large and must be optimized for a given problem. A constant weight decay parameter has been used throughout a training session.

Rank Regression. Pleiss, examining the steroid data, pointed out, and we confirm, that π_b and SAO are not normally distributed. Therefore, he used rank regression. The parameter set he obtained that best described the data was $\{R(\text{SAO}), R^2(\text{SAI}), R(\pi_a)\cdot R(\text{SAO}), R(\pi_b)\cdot R(\text{SAO}), R(\text{MK})\cdot R(\text{CC})\}$.

Here "R(name)" is the rank of the variable "name". This equation gives $\rho = 0.91$ and $s = 7.27$, resulting in a better model. It was used to predict the activities of the 10 compounds not included in its derivation and 4 compounds from the training set. Out of the 14 compounds 12 were predicted better by the rank regression method. The other two compounds yield very large residuals (unlike Lee *et al.*). Pleiss rationalized this by observing that these two compounds (62 and 65 in Table IX) have an 18-ethyl group which is *not* present in any other compound in the database on which the regression equation was developed. He therefore concluded that this moiety is not well parameterized.

Results and Discussion

Triazine. From the modeling standpoint we focus on two issues: (1) the new architecture that separates out the linear and nonlinear influences on biological activity and (2) on demonstrating the crucial importance of weight decay for better convergence.

Graphical analysis shows that the distribution of σ deviates significantly from normal. This is apparent from the histogram in Figure 3. For comparison, we have also shown the histogram of π which is roughly normal.

Fitting Linear Hypersurfaces Using a Neural Network. The first step in building QSARs is to try a linear model. Using the standard architecture we are never sure whether the sigmoidal units are being used to fit straight lines or not. Using a perceptron with linear output units, on the other hand, makes the step of fitting straight lines clear and simple. If a perceptron is not able to model the data, we can add hidden (sigmoidal) units till we obtain a satisfactory model. To facilitate the use of neural networks as a standard QSAR modeling tool, we have to demonstrate the empirical equivalence of perceptrons with linear units to multiple linear regression.[11]

The perceptron results are given in Table I (the top six rows). They indicate large errors, which is analogous to what Selassie *et al.* found. Next, following a common procedure in QSAR modeling, we introduce a nonlinearity in π using $\log(\beta 10^{\pi'} + 1)$ as an extra input. The results for this are also shown in Table I. As expected, our results are very similar to the ones obtained by Selassie *et al.* We use the value of β that was obtained in their work for the results shown in the table. Its value can also be determined by using, for example, an iterative approximation scheme. The three "outliers" found by Selassie *et al.* are the same compounds the perceptron deems as "outliers". The equation describing biological activity obtained using a perceptron is very similar to the one obtained by Selassie *et al.* This demonstrates the empirical equivalence we set out to show.

Table I[a]

parameters	d	ρ	regression
π, MR	0.55	0.16	0.98 ± 0.79; 0.14 ± 5.16
π', MR	0.48	0.50	0.99 ± 0.22; 0.01 ± 1.47
σ, π	0.55	0.16	1.00 ± 0.78; −0.03 ± 5.14
σ, π'	0.49	0.49	0.99 ± 0.23; 0.00 ± 1.49
σ, π, MR	0.55	0.19	0.99 ± 0.65; 0.01 ± 4.25
σ, π', MR	0.48	0.51	0.99 ± 0.22; 0.00 ± 1.45
π', $\log(\beta 10^{\pi'} + 1)$	0.33	0.80	0.99 ± 0.09; 0.02 ± 0.63
π', $\log(\beta 10^{\pi'} + 1)$, MR	0.31	0.83	1.00 ± 0.08; −0.04 ± 0.57
π', $\log(\beta 10^{\pi'} + 1)$ σ	0.30	0.84	1.00 ± 0.09; 0.00 ± 0.54

[a] The results of using a Perceptron on learning the mapping for 3-X-triazine inhibition of purified DHFR from L1210 leukemia cells resistant to methotrexate are shown. The first column shows the parameters used in learning. d is the summed squared deviation and ρ the correlation coefficient between the observed and fitted biological activities. The last column shows the regression coefficients (slope and intercept, respectively) of plotting a line through the calculated and observed activities. All 61 points were used. As is obvious, including the log term improves the results drastically. Eliminating the points that were the hardest to fit (the same as the ones removed from the analysis by Selassie *et al.*; see text) and using π' and $\log(\beta 10^{\pi'} + 1)$, we get $d = 0.87$ and $\rho = 0.29$—results comparable to Selassie *et al.* Our best determination of β was 0.1, close to the number they obtain. Hence we note that the conclusions of multiple linear regression are reproduced.

Table II[a]

parameters	d	ρ	regression
π, MR	0.38	0.73	1.00 ± 0.19; 0.00 ± 0.76
π', MR	0.30	0.84	1.00 ± 0.06; −0.20 ± 0.40
σ, π	0.37	0.74	1.00 ± 0.18; 0.00 ± 0.75
σ, π'	0.43	0.63	1.00 ± 0.16; 0.00 ± 1.04
σ, π, MR	0.37	0.74	1.00 ± 0.18; 0.00 ± 0.77
σ, π', MR	0.35	0.78	0.99 ± 0.10; 0.00 ± 0.69

[a] The results of using 1 hidden unit to learn the mapping for 3-X-triazine inhibition of purified DHFR from L1210 leukemia cells resistant to methotrexate are shown. The first column shows the parameters used in learning. d is the summed squared deviation and ρ the correlation coefficient between the observed and fitted biological activities. The last column shows the regression coefficients (slope and intercept, respectively) of plotting a line through the calculated and observed activities. All 61 points were used. Networks were constructed with no weight decay. Out of 30 different runs (using different seeds to initialize the networks), only a few gave good results (the actual number varied for the different parameters). Other runs got stuck in spurious minima.

The best results are obtained by using π' rather than π. Furthermore, σ and MR seem equally relevant. Weight decay was not useful for perceptron learning.

Adding Nonlinear Hidden Units. One advantage of using neural networks is that nonlinear dependencies can be automatically incorporated. Therefore we now add sigmoidal hidden units, one at a time, to the perceptron instead of the $\log(\beta 10^{\pi} + 1)$ input. Thirty different networks, with and without weight decay for different combinations of input parameters, were studied at each stage (see Tables II and III for results).

For the parameters in the first three rows (of Table III) the results do not change on introducing weight decay. Unlike the case with zero weight decay, however, all 30 runs converge and none get stuck in spurious minima. For the parameters in the last four rows, on the other hand, we note a substantial improvement in both the correlation coefficient and standard deviation, but this time, less than 5% of the runs get stuck in spurious minima. This demonstrates the utility of weight decay, though the benefits are different for different networks. From Table II (zero weight decay) it appears that {π', MR} yields the best model. On the other hand the results in Table III demonstrate that there are other parameter choices ({π, MR}, {σ,π'}, and {σ, π', MR}) that yield equivalent results. This conclusion is reaffirmed by a visual inspection of the three models. Therefore, consideration of weight decay prevents us from drawing incorrect conclusions about the data set.

Table III[a]

parameters	d	ρ	regression	decay
π, MR	0.38	0.73	0.99 ± 0.20; 0.00 ± 0.70	0.001
π'	0.33	0.81	1.00 ± 0.18; 0.10 ± 0.50	0.001
π', MR	0.30	0.84	1.01 ± 0.10; 0.20 ± 0.50	0.001
σ, π	0.37	0.74	0.99 ± 0.25; 0.00 ± 0.68	0.001
σ, π, MR	0.37	0.75	1.00 ± 0.11; −0.05 ± 0.75	0.001
σ, π'	0.29	0.85	1.03 ± 0.08; −0.21 ± 0.55	0.001
σ, π', MR	0.27	0.87	1.01 ± 0.07; −0.07 ± 0.49	0.001

[a] The results of training a network to learn the mapping for 3-X-triazine inhibition of purified DHFR from L1210 leukemia cells resistant to methotrexate under similar conditions as in Table II except for a nonzero weight decay are shown. A constant weight decay parameter is used from the start of learning and the optimum value is shown.

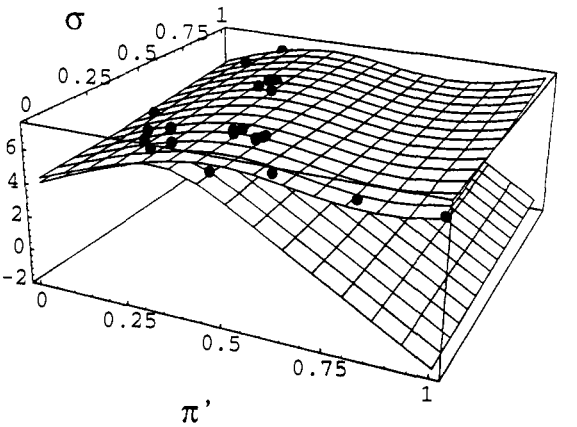

Figure 6. The results for inhibition of purified DHFR from leukemia cells resistant to methotrexate. This compares the neural network model and the Selassie *et al.* model. The Selassie *et al.* model is roughly parabolic, and a comparison points out the flexibility of a neural network.

Two of the better networks with weight decay are

$$O = 4.08 + 1.54\sigma + 10.12\pi' - \frac{8.90}{1 + \exp[4.22 - 0.81\sigma - 6.30\pi']} \quad (5)$$

and

$$O = 4.94 + 7.99\pi' - 0.54\mathrm{MR} - \frac{6.76}{1 + \exp[4.60 - 6.78\pi' - 0.15\mathrm{MR}]} \quad (6)$$

Figure 6 compares the fits obtained by Selassie *et al.* and a neural network with one hidden unit. This figure clearly shows why Selassie *et al.* had to drop three points while building their regression equation.

Next, we build networks with two units in the hidden layer. Neither the statistical measures nor visual inspection of the surfaces shows any improvements. There are a few more kinks and wrinkles on the surface when two hidden units are used. This is an attempt by the network to adjust finely to the distribution of points.

In effect, we have two basic conclusions from our networks. First, there is no reason to prefer {π', MR} over {π', σ}. Both give effectively the same cross-correlation coefficient and the sum of the squared difference between the calculated and observed activities. There are two important points with regard to using an equation involving

Table IV[a]

parameters	d	ρ	decay	d	ρ	decay
π, MR	0.29	0.81	0.0	0.27	0.84	0.001
π'	0.43	0.48	0.0	0.36	0.69	0.001
π', MR	0.32	0.76	0.0	0.30	0.80	0.001
σ, π	0.31	0.78	0.0	0.31	0.78	0.001
σ, π, MR	0.23	0.89	0.0	0.23	0.89	0.001
σ, π'	0.32	0.77	0.0	0.32	0.77	0.001
σ, π', MR	0.29	0.81	0.0	0.29	0.81	0.001

[a] The results of using two hidden units on learning the mapping for 3-X-triazine inhibition of cultured L1210 leukemia cells sensitive to methotrexate are shown. Results obtained without and with weight decay are shown. The weight decay parameters shown have been optimized.

Table V[a]

parameters	d	ρ	decay	d	ρ	decay
π, MR	0.25	0.87	0.0	0.24	0.88	10^{-4}
π'	0.36	0.69	0.0	0.36	0.68	10^{-4}
π', MR	0.27	0.83	0.0	0.29	0.81	10^{-4}
σ, π	0.29	0.81	0.0	0.29	0.81	10^{-4}
σ, π, MR	0.19	0.92	0.0	0.19	0.92	10^{-4}
σ, π'	0.29	0.80	0.0	0.29	0.80	10^{-4}
σ, π', MR	0.25	0.86	0.0	0.25	0.86	10^{-4}

[a] The results of using three hidden units on learning the mapping for 3-X-triazine inhibition of cultured L1210 leukemia cells sensitive to methotrexate are shown. Results obtained without and with weight decay are shown. The weight decay parameters shown have been optimized.

either MR or σ. The correlation coefficient between π' and MR is small ($\rho^2 = 0.26$) and so is the one between π' and σ ($\rho^2 = 0.28$). Therefore, as far as collinearity is concerned, both parameters are equivalent. We need to keep in mind that the F-test is *not* a valid reason to choose σ over MR because, as noted, σ is not normally distributed. However we confirm the preference for π' over π.

3-X-Triazine Inhibition of Cultured L1210 Leukemia Cells Sensitive to Methotrexate. Unlike Selassie *et al.*, we do not use indicator variables for this data set. The perceptron results (not shown) follow along the same lines as before. Weight decay is again seen, from Tables IV and V, to be important for learning with lower error and higher correlation coefficient. A clear preference is seen for π over π' in all cases.

The network using two hidden units and weight decay show that there is a preference for $\{\pi, \text{MR}\}$ over $\{\pi, \sigma\}$. Again, σ cannot be retained or rejected based on the F-statistic. The linear correlation between π and MR is rather high ($\rho^2 = 0.6$) compared to that between π and σ ($\rho^2 = 0.2$) and σ and MR ($\rho^2 = 0.24$). Therefore, care must be employed when using the $\{\pi, \text{MR}\}$ data for extrapolations.

The best equation obtained using two hidden units and weight decay is

$$O = 0.77 - 0.74\pi - 1.45\text{MR} + \frac{7.62}{1 + \exp[3.36 - 6.09\pi - 6.58\text{MR}]} + \frac{5.43}{1 + \exp[-3.92 + 0.05\pi + 18.34\text{MR}]} \quad (7)$$

This can be simplified by dropping the 0.05π term. A visual examination of the different equations also shows that the $\{\pi, \sigma\}$ network has more outliers than the $\{\pi, \text{MR}\}$ network. The large linear correlation coefficient between π and MR is reflected in the "flatness" of the plot. Since the extent of the flat region is known, careful extrapolation on new data is feasible.

Table VI[a]

parameters	d	ρ
π	0.29	0.89
π, MR	0.24	0.93
π', MR	0.53	0.58
σ, π	0.28	0.90
σ, π, MR	0.24	0.93
σ, π'	0.58	0.44
σ, π', MR	0.53	0.58

[a] The results of using a perceptron on learning the mapping for 3-X-triazine inhibition of cultured L1210 leukemia cells resistant to methotrexate are shown. There are no benefits (or losses) when using weight decay with perceptrons.

Table VII[a]

param set	ρ (train)	error (train)	ρ (test)	error (test)	decay
a	0.96	0.30	0.92	1.61	0.0001
a	0.96	0.32	0.94	1.32	0.001
a	0.96	0.30	0.92	1.61	0.01
a	0.96	0.29	0.91	1.84	0.0
b	0.89	0.51	0.90	1.42	0.001
b	0.89	0.51	−0.81	9.01	0.0

[a] A sample of the results on the train and test sets for the steroid data is shown. Set a uses π_a, π_b, SAI, SAO, MK, and CC as input parameters; and set b uses π_b, SAO, MK, and CC. Notice the significant change in performance on the test set while the performance on the train set is about the same, when we include weight decay. In particular note the negative correlation coefficient in the last row for the test set performance using no weight decay.

Note that the results from the two tables do not conclusively indicate the best model, for example, $\{\sigma, \pi, \text{MR}\}$ with two hidden units is equivalent to $\{\pi, \text{MR}\}$ with three hidden units. Using a large number of parameters improves the fit as before. The only way to decide on the best model is to examine the respective generalization abilities. We will come back to this point later in the paper when we consider building networks for the steroid data (see also the discussion in the first paragraph of Methods).

3-X-triazine Inhibition of Cultured L1210 Leukemia Cells Resistant to Methotrexate. A perceptron was again trained, with $\{\pi, \text{MR}\}$ yielding the best network. The results are given in Table VI. Attempts to build networks using hidden units did not yield improvements. This is a nice result because it demonstrates that the fit cannot always be improved by adding hidden units when a linear equation is the best possible model. Therefore, a linear relationship is the best one for leukemia cells resistant to methotrexate. As mentioned, π and MR have a large linear correlation. Once again, we must be cautious with extrapolations.

Steroid. The steroid data set is analyzed to compare multiple linear regression, rank regression, and neural network regression. Here we concentrate on building models that are good at generalization.

We developed networks with many combinations of input parameters using the complete training set. A sample of the results is shown in Table VII. As the test set performances indicate, nonzero weight decay networks yield a better model (significantly increased correlation coefficient and decreased squared-error) predictions even when a sloppy training procedure that does not perform any cross-validation is adopted.

Of course, building models that minimize the error in the training set is not the best method to adopt for robust modeling. Many parameters like the weight decay parameter, the number of hidden units, and inputs that define

Table VIII[a]

param set	CVE	corr coeff	decay
a	0.98	0.53	0.0001
b	0.77	0.75	0.0001
c	0.47	0.91	0.001
d	0.95	0.62	0.0005
e	0.65	0.81	0.0001
f	0.58	0.86	0.0001
g	0.61	0.83	0.01

[a] This table shows the results of cross-validation runs for the steroid data set. We only show a sample of the different possibilities examined in this study (the better ones). The first column indicates the different input parameters used. Set a includes π_a, π_b, SAI, and, SAO; b includes π_a, π_b, SAI, SAO, and MK; c includes π_a, π_b, SAI, SAO, MK, and CC; d includes π_a, SAI, MK, and CC; e includes π_b, SAI, MK, and CC; f includes π_b, SAI, SAO, MK, and CC; and set g includes π_b, SAO, MK, and CC. CVE is the cross-validation error. "Corr coeff" is the correlation between the predicted and the observed biological activities for the validation set. Finally the last column shows the weight decay parameter required to obtain the best network in each case. All of these use two hidden units. Using either one or three hidden units (with and without weight decay) increases CVE.

the network have to be optimized. One of the oldest methods in statistics that achieves good performance is cross-validation. Cross-validation is a useful technique because *in practice* it has often been found to correlate well with generalization error. The cross-validation error for network j is interpreted as an estimate of the generalization error of network j when trained on the complete data set.

We adopt a J-fold cross-validation procedure with J = 11. Here the test sets have 5 nonduplicated compounds out of a total of 55 in the complete data set. We train many networks with various combinations of input parameters, hidden units, and weight decay parameters. A sample of our results is shown in Table VIII. The cross-validation error (CVE) is calculated for each network by using, CVE = $\sum_{i=1}^{N} E_{i,v}^2/N$, where $E_{i,v} = N_{i,v}^{out} - A^{out}$. $N_{i,v}^{out}$ represents the output predicted by the ith network on the validation set compounds, and A^{out} is the observed output. The model (network with a particular set of weights and connectivity) that gives the smallest CVE is used for further predictions.

The most interesting aspect of Table VIII is that the best results are obtained by networks that incorporate a weight decay term. This is true immaterial of the choice of input parameters or the number of hidden units used. A comparison of cross-validated error of the best network (set c—{π_a, π_b, SAI, SAO, MK, and CC} and two hidden units), and the maximum error on the test set shows that CVE yields a resaonable upper-bound estimate for the generalization error for this data set. (Of course, CVE is not an upper bound for any arbitrary generalization—only reasonable generalizations. It is also true that the particular value of J chosen for cross validation experiments is important. For example, there are data sets where a 1-fold cross validation is inferior to other methods for cross validation.) Both the neural network model and rank transform regression result in a similar correlation coefficient.

It is instructive to compare networks with and without weight decay and with different hidden units. The results for sets c and g provide typical examples. The best network with no weight decay (two hidden units) yielded a CVE of 0.54 and 0.68 for sets c and g respectively. The corresponding network using the optimum weight decay yields an improvement of ≈15%. For certain series like

Table IX[a]

no.	obsd	calcd	% error	calcd	% error	calcd	% error
1	2.0	1.87	5.0	1.99	0.5	2.13	−6.5
27	2.41	2.18	8.3	2.37	1.7	2.63	−9.0
32	2.23	1.76	22.0	1.76	21.1	2.27	−1.8
47	2.62	1.99	25.2	1.99	24.0	2.45	6.5
57	2.00	1.43	27.0	1.83	8.5	1.41	29.5
58	1.30	0.68	44.6	1.55	−19.2	1.66	−27.7
59	2.70	2.45	8.5	2.62	3.0	2.42	10.3
60	2.30	1.70	25.2	2.08	9.6	1.77	23.0
61	1.74	0.95	55.2	1.87	−7.5	2.07	−18.9
62	1.54	1.20	22.1	0.30	80.5	0.50	67.5
63	1.30	0.27	80.8	1.24	4.6	1.19	8.5
64	1.30	0.50	63.8	1.37	−5.4	1.97	−51.5
65	0.70	−0.29	141.4	−0.94	234.3	−0.44	162.9
66	2.48	2.02	16.1	2.30	7.3	1.68	32.2

[a] The comparative results obtained by Lee *et al.*, Pleiss, and in this work for the test set are shown (this explores generalization abilities).

the steroid data and unlike the triazines, depending on the parameterization and noisiness of the data, different initial networks result in significantly different models. The CVEs of these models fluctuate by large amounts from the average (we designate the "network" that yields average CVE as a "typical network"). These fluctuations point to the existence of many local minima. The typical network show an improvement of ≈30% when weight decay is used. Also the fluctuations among the models developed with weight decay are smaller than the ones found when no weight decay is used, implying that weight decay flattens some of the spurious local minima.

Analogously, the CVE values for the best network with one hidden unit are 0.5 with an optimum *decay* of 0.01, and 0.64 with an optimum *decay* of 0.01 for sets c and g, respectively. These results imply that networks with two hidden units improve the CVE by ≈5–6%.

We now have a trained network that yields the lowest CVEs which can be used as the model for the data. Our predictions using this model are shown in Table IX. It performs much better than multiple linear regression. (As in triazines, the perceptron is found to be equivalent to multiple linear regression.) We, like Pleiss, identify the 18-ethyl moiety as the one that yields the largest errors. In addition, the 4,9,11-estratriene moiety results in intermediate errors.

An important point must be mentioned about cross-validation. One of the biggest problems with cross-validation is that it never uses the entire training data. A method to tackle this and the associated multiple-minima problem that is often encountered with cross-validation, is addressed in a forthcoming paper.[9]

Conclusion

Compared to multiple linear regression, application of neural networks to function approximation is quite new. We have demonstrated that consideration of some crucial details that have so far been neglected enhances the applicability of neural networks to quantitative structure–activity relationships.

The equivalence of a perceptron to multiple linear regression, linear output units, direct input–output connections, and the ability to add one hidden unit at a time to the perceptron, facilitates the use of neural networks as a common tool for building QSARs. The linear output units and direct input–output connections also help in the interpretation of the models as the resulting equations are simpler.

Weight decay is shown to be important both for learning *and* generalization. It helped us build a model for the triazine data which indicates that the $\{\pi', \sigma\}$ network is as good as the $\{\pi', MR\}$ network for the inhibition of purified DHFR. This conclusion is different from that obtained by Selassie *et al.*[5] Weight decay is also important for the steroid data as it yields lower cross-validated errors. We find weight decay to be so important that it should always be considered. Without it, one can never be sure whether or not the network under consideration provides the best possible model for the data. Furthermore, interpretations and extrapolations cannot be made reliably. This is amply demonstrated both from the results in this work and our experience from other data sets.

As the steroid results indicate, the non-normality of data is not a constraint for neural networks. Some of the essential characteristics of the steroid data are brought out by neural network modeling. We have also shown the usefulness of cross-validation for settling on the best possible network (*i.e.*, inputs, hidden units, weight decay, and in principle, other parameters).

The distribution of points in a data set that uses the original variables (as in neural networks) is very different from the same data set that uses ranks of the original variables (as in rank regression). Hence it is not appropriate to directly compare their usefulness to QSAR. Without further theoretical and empirical study, we cannot conclude that neural networks can be a replacement for other nonparametric regression techniques or *vice versa*. Additionally, rank regression itself could equally well have been carried out using a neural network just as for the real data, with less effort than Pleiss.

Since linear models have worked well and neural networks are relatively computer intensive, there is a reluctance to its widespread use in the QSAR community.[12] The use of the standard architecture in Figure 4, and the slow standard backpropagation procedure reinforces these reasons, respectively. The architecture proposed in this work, on the other hand, uses perceptrons to fit linear models and adds nonlinearities only when needed, while maintaining the separation of linear and nonlinear contributions. We also use the quasi-Newton method and weight decay to speed up convergence. The effort required to search through the weight decay parameter space is not large because only small values are useful and a grid search is sufficient. In fact, typical networks built using any small randomly chosen weight decay yields better models than networks with no weight decay for both the data sets. Though this may not be true for all data sets. It also improves convergence and results in faster training of networks.

Coming to questions of computational effort we note the following. Perceptron training is as easy and quick as multiple linear regression.[13] However, if a perceptron is not found to be appropriate, hidden units can profitably be used. This is because it is too time consuming to go through each possible nonlinearity. In our experience, the effort required to determine the weights of a single hidden unit network is comparable to that of calculating the value of β in the familiar $\log(\beta 10^{r} + 1)$ nonlinearity. In terms of CPU cycles, for the largest network, training with weight decay took about 45 s, and without weight decay about 55 s on a DEC 5100 using an in-house program written in C. A typical one hidden unit network, on the other hand, takes about 20 s of CPU time. Only cross-validation, required for statistical soundness and which is necessary for both neural networks and multiple linear regression, requires a large amount of effort.

Acknowledgment. I would like to thank Prof. Robert Langridge for providing constant encouragement while this work was in progress. I am grateful to Keven Clark and Leslie Taylor for comments on an earlier draft of this paper. I would like to acknowledge the invaluable assistance of Poonam Pillai in making this paper readable. This work was supported by research grant RR-1081 from the National Center for Research Resources, NIH.

Supplementary Material Available: Tables of the triazines and steroids used in the study with the physicochemical parameters and biological activity (7 pages). Ordering information is given on any current masthead page.

References

(1) Andrea, T. A.; Kalayeh, H. Applications of neural networks in quantitative structure-activity relationships of dihydrofolate reductase inhibitors. *J. Med. Chem.* **1991**, *34*, 2824–2836.
(2) So, S. S.; Richards, W. G. Application of neural networks: quantitative structure-activity relationships of the derivatives of 2,4-diamino-5-(substituted-benzyl)pyrimidines as DHFR inhibitors. *J. Med. Chem.* **1992**, *35*, 3201–7.
(3) Ajay. A New Neural Net Training Algorithm for QSAR. Preprint, Dept. of Pharmaceutical Chemistry, University of California, San Francisco.
(4) Pleiss, M. A. The Application of Nonparametric Statistical Methodology to Quantitative Structure Activity Relationship (QSAR) Studies. In *QSAR in design of bioactive compounds*; Kuchar, M., Ed.; J. R. Prous International Publishers: Barcelona, Spain, 1984; pp 403–424.
(5) Selassie, C. D.; Strong, C. D.; Hansch, C.; Delcamp, T. J.; Freisham, J. H.; Khwaja, T. A. Comparison of Triazines as Inhibitors of L1210 Dihydrofolate Reductase of L1210 Cells Sensitive and Resistant to Methotrexate. *Cancer Res.* **1986**, *46*, 744–756.
(6) Lee, D. L.; Kollman, P. A.; Marsh, F. J.; Wolff, M. E. Quantitative Relationships between Steroid Structure and Binding to Putative Pregesterone Receptors. *J. Med. Chem.* **1977**, *20*, 1139–1146.
(7) Hertz, J.; Krogh, A.; Palmer, R. G. *Introduction to the Theory of Neural Computation*; Addison Wesley: Redwood City, CA, 1991.
(8) Press, W. H.; Flannery, B. P.; Teukolsky, S. A.; Vetterling, W. T. *Numerical Recipes*; Cambridge University Press: Cambridge, 1986.
(9) Ajay. On Better Generalization by Combining Two or More Neural Networks—a QSAR Example. Preprint, Dept. of Pharmaceutical Chemistry, University of California, San Francisco.
(10) Strictly speaking it is not an indicator variable. Indicator variables in regression should take on only two values, 1 or 0. This is not the case for CC. We use it as a normal parameter (not an indicator variable) that can be scaled.
(11) This is important as we are not aware of any such explicit demonstration elsewhere.
(12) I thank the reviewers for bringing up this point.
(13) One advantage of multiple linear regression over the often-used neural networks is that it is relatively easier to assign error bars to the predictions.