Neural Network Studies. 3. Variable Selection in the Cascade-Correlation Learning Architecture

Vasyl V. Kovalishyn,[†,‡] Igor V. Tetko,*[,†,§,||] Alexander I. Luik,[†] Vladyslav V. Kholodovych,[†] Alessandro E. P. Villa,[§] and David J. Livingstone[⊥]

Institute of Bioorganic and Petroleum Chemistry, Ukrainian Academy of Sciences, Murmanskaya, 1, Kiev-660, 253660, Ukraine, and Laboratoire de Neuro-Heuristique, Institut de Physiologie, Université de Lausanne, Rue du Bugnon 7, Lausanne, CH-1005, Switzerland, and ChemQuest, Cheyney House, 19-21 Cheyney Street, Steeple Morden, Herts, SG8 0LP, U.K., and Centre for Molecular Design, University of Portsmouth, Portsmouth, Hants, PO1 2EG, U.K.

Received January 26, 1998

Pruning methods for feed-forward artificial neural networks trained by the cascade-correlation learning algorithm are proposed. The cascade-correlation algorithm starts with a small network and dynamically adds new nodes until the analyzed problem has been solved. This feature of the algorithm removes the requirement to predefine the architecture of the neural network prior to network training. The developed pruning methods are used to estimate the importance of large sets of initial variables for quantitative structure−activity relationship studies and simulated data sets. The calculated results are compared with the performance of fixed-size back-propagation neural networks and multiple regression analysis and are carefully validated using different training/test set protocols, such as leave-one-out and full cross-validation procedures. The results suggest that the pruning methods can be successfully used to optimize the set of variables for the cascade-correlation learning algorithm neural networks. The use of variables selected by the elaborated methods provides an improvement of neural network prediction ability compared to that calculated using the unpruned sets of variables.

INTRODUCTION

Recently there has been a growing interest in the application of neural networks in the field of quantitative structure−activity relationships (QSARs). It has been demonstrated that this technique is often superior to the traditional approaches.[1−3] Artificial neural networks are able to perform nonlinear mapping of the physicochemical descriptors to the corresponding biological activity implicitly and can overcome some limitations of traditional QSAR approaches.

The rational search of parameters describing some biological activity of molecules requires information about molecular structures and physicochemical properties. For such information researchers use a number of physicochemical descriptors (hydrophobicity, steric and electronic parameters, etc.). As a result a relatively large number of input variables can be available for analysis. The basic question raised is how to determine the most relevant variables. Traditional methods, such as multiple linear regression (MLR), provide a tool for reducing nonrelevant variables.[4] However, this problem has only been partially investigated for back-propagation neural networks. Several pruning algorithms for determination of the relevance of input variables have been proposed and investigated,[5−8] and successful application of these algorithms has been reported.[9−12] The neural networks used in the previous studies were characterized by fixed-size architectures; i.e., the number of hidden neurons, the number of connection weights, and the connectivity amid layers were all fixed. It is known that the performance of neural networks is fundamentally tied to their topology. The capacity and accuracy of a network mapping is determined by the number of free parameters (typically weights) in the network. Neural networks that are too small (underfitting) cannot accurately approximate the desired input-to-output mapping, while too large networks can have a lower generalization ability because of the overfitting/overtraining problem[13−15] and require longer training time. The selection of architectures in such networks is usually done on the basis of trial/test results. However, an incorrect selection of the topology can decrease the performance of the method.

There are some other algorithms, so-called topology-modifying[16] neural network algorithms, that are able to automatically determine an optimal neural network architecture that is pertinent to the analyzed problem. These algorithms start with a small network and add weights or/and nodes until the problem has been solved. Thus, they do not require the determination of the size of neural networks prior to learning and are not subjected to the problem of underfitting. These topology-modifying algorithms are of considerable interest for practical applications because of their ability to solve some tasks (e.g., the two-spirals problem[17]) which represent substantial difficulties for the training of fixed-size neural networks. In addition, the topology-modifying algorithms are considerably faster compared to fixed-size neural networks.[18]

[†] Ukrainian Academy of Sciences.
[‡] E-mail: vkov@bioorganic.kiev.ua.
[§] Université de Lausanne.
[||] E-mail: itetko@eliot.unil.ch.
[⊥] ChemQuest and University of Portsmouth. E-mail: davel@chmqst.demon.co.uk.

The current study introduces pruning algorithms for one of the most popular topology-modifying algorithms—the cascade-correlation neural network (CCN). We show that the optimization of input variables increases the overall performance of this algorithm for simulated data sets and real QSAR examples.

AVOIDANCE OF OVERFITTING/OVERTRAINING PROBLEM

The avoidance of overfitting/overtraining has been shown to be an important factor for the improvement of generalization ability and correct selection of variables in fixed-size back-propagation neural networks (BNN)[15] and in CCN.[18] The early stopping over ensemble (ESE) technique was used in the current study to accomplish this. A detailed description of ESE can be found elsewhere.[15,18,19] In brief, each analyzed artificial neural network ensemble (ANNE) was composed of $M = 100$ networks. The values calculated for analyzed cases were averaged over all M neural networks, and their means were used for computing statistical coefficients with targets. We used a subdivision of the initial training set into two equal learning/validation subsets.[20] The first set was used to train the neural network, while the second one was used to monitor the training process. Two stopping points were used to test network performance measured by root-mean-square error (RMSE). The first point (early stopping) determined a best fit of a network to the validation set, while the second point corresponded to the error minimum for the learning set and, as a rule, coincides with the end of the network training.[15] The correlation coefficient R and cross-validated q^2 value[15,21] calculated by the leave-one-out (LOO) method[15] were used as a measure of the predictive ability of the networks. The training was terminated by limiting the network run to 10 000 epochs (total number of epochs) or after 2000 epochs (local number of epochs) following the last improvement of RMSE in the early stopping point. The training of each network in the ensemble was done as described in the next section.

CASCADE-CORRELATION LEARNING ARCHITECTURE

The CCN was proposed by Fahlman and Lebiere.[17] The cascade architecture contains input and one or more output neurons, and some hidden neurons are added in the training time (Figure 1). The number of inputs and outputs are conditioned by the task. Every input and output is connected through an adjustable weight value w_{ij}. There is also a bias input, permanently set to +1. The logistic $f(x) = 1/(1 + e^{-x})$ activation function was used in our study both for hidden and output nodes.

The algorithm begins the training with no hidden neurons. The hidden neurons are added to the network one by one. Each new hidden neuron receives a connection from all of the network's original inputs and also from the preexisting hidden neurons. For any new hidden neuron the magnitude of the correlation between the new neuron's output and the residual error signal is eliminated, as described in ref 17. The input weights of each hidden neuron are frozen after the neuron has been added to the network. Therefore each new neuron adds a new one-neuron "layer" to the network. After addition of a new hidden neuron the output connections

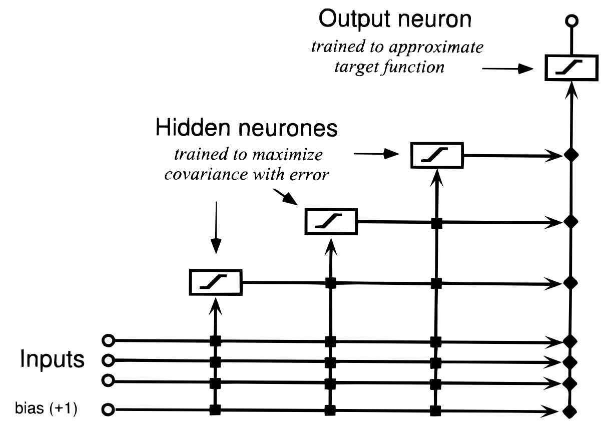
Output neuron
trained to approximate target function
Hidden neurones
trained to maximize covariance with error
Inputs
bias (+1)

Figure 1. Topology of an artificial neural network with cascade-correlation architecture. There are three inputs, three hidden neurons, and one output. The vertical lines sum all incoming activation. Boxed connections are frozen; diamond connections are trained repeatedly.

are retrained. We limited the maximal size of hidden layers to 50 neurons. This number was enough to train CCN and to determine the early stopping points for all of the analyzed data sets.

The "quickprop" algorithm[22] was used to train the output weights. The performance of each neural network for the validation set was continuously monitored during this step of training. At some point, the training became asymptotic, and it was stopped if there was not a significant reduction of RMSE (less than 0.001) after 50 epochs. The criteria for termination of network training were checked (i.e., total or local number of epochs, maximal number of hidden layers), and if at least one such criterion was exceeded, the training was terminated; otherwise we attempted to reduce the residual errors by adding a new hidden neuron to the network and to retrain output weights as described above. More details concerning this algorithm can be found in ref 17.

ANALYZED PRUNING METHODS

A great number of pruning methods for neural networks have been proposed in recent years.[23−28] These methods can be divided into two main groups: (1) sensitivity methods and (2) penalty term methods.[28] The methods of the first group are the most appropriate for determination of the relevancy of input variables. These methods introduce some measures of weight importance by so-called sensitivities. The sensitivities of all weights and input nodes are estimated, and elements with the smallest sensitivities are deleted.[23−25] Our previous studies analyzed several pruning methods developed for BNN.[5,6,9] The architecture of CCN significantly differs from that of fixed-size neural networks. The main differences are the dynamic addition of new hidden nodes during network training and the different type of neural connectivity in the network. We investigated if the same general principles of estimation of the sensitivity of input variables that have been used successfully for BNN could be also applied to CCN.

The topology of cascade correlation is composed in such a manner that each neuron of the input layer is connected to all neurons of the hidden and output layers. A sensitivity S_i of input variable i is introduced by

$$S_i = \sum_{k \in \Omega_i} s_k \equiv \sum_{j=1}^{n_j} s_{ji} + \sum_{m=1}^{n_m} s_{mi} \tag{1}$$

where s_k is the sensitivity of the weight w_k and summation is over a set Ω_i of outgoing weights of the neuron i or, using another order of weight numeration, s_{ji} is the sensitivity of weight w_{ji} connecting the ith neuron to the jth neuron in the hidden layer, s_{mi} is the sensitivity of weight w_{mi} connecting the ith neuron and the mth neuron in the output layer, and n_j (n_m) is the number of neurons in the hidden (output) layer. The same formula is applied to estimate the sensitivity of a hidden neuron except summation is restricted to the hidden neurons that were added after the analyzed neuron. Equation 1 is the same as that used for BNN[6] except the number of neurons in the hidden layer is determined by the training procedure of CCN, and there is an additional term corresponding to the direct connections of input neurons to the output layer.

In our previous study[6] five pruning methods, designated as A−E, were developed for analysis of fixed-size neural networks. Three of these methods, namely, A, B, and D, were programmed for CCN, while the two other algorithms were not. Method C was excluded since, as reported elsewhere,[6] its ability to generalize is poor. Method E was also excluded because of the significant amount of computing time required (the speed of this method decreases as a function of z^2, where z is the number of weight connections and the number z is much bigger for CCN than in BNN).

The first two methods A and B are called "magnitude-based methods" because a neuron's sensitivity is based on direct analysis of the magnitudes of its outgoing weights. The sensitivity in the third method D is defined by a change of the network error due to the elimination of some neuron weights ("error-based method").

Method A. The first method explores the idea that input neurons with bigger connection weights to the hidden and output layers play a more significant role than other input neurons. Therefore, the absolute magnitude of weight w_k

$$s_k = |w_k| \tag{2}$$

is used as its sensitivity in eq 1.

Method B. The sensitivity in the second method for BNN is calculated by

$$S_i = \sum_{j=1}^{n_j} \left(\frac{w_{ji}}{\max_a |w_{ja}|} \right)^2 S_j \tag{3}$$

where $\max_a$ is taken over all weights ending at neuron j, S_j is the sensitivity of the jth neuron in the upper layer, and n_j is the number of hidden neurons. The sensitivities of output layer neurons are set to 1. The equation for the cascade correlation topology is very similar

$$S_i = \sum_{j=1}^{n_j} \left(\frac{w_{ji}}{\max_a |w_{ja}|} \right)^2 S_j + \sum_{m=1}^{n_m} \left(\frac{w_{mi}}{\max_a |w_{ma}|} \right)^2 \tag{4}$$

except there is an additional term accounting for direct connections of the neuron i to the outputs m. The calculation of neuron sensitivities by this formula is done in a recurrent way starting from the neurons at the highest hidden layers; i.e., the neurons added to the network at the latest cycles of training.

Method D. Optimal brain damage[23] uses the second derivative of the error function with respect to the neuron weights to compute the sensitivities as follows:

$$s_k = (w_k)^2 (\partial^2 E / \partial w^2_k) \tag{5}$$

This weight sensitivity is used in eq 1.

SENSITIVITY SCORE OF INPUT NEURON

The sensitivities of all input neurons were determined for each neural network following ensemble analysis. Unless otherwise noted, the sensitivities were calculated at the early stopping point, which corresponded to the minimum of RMSE for the validation data set. For each input, we counted the number of neural networks for which it had the lowest sensitivity. This number was considered as a score for the input neuron sensitivity in the ensemble. The input characterized by the biggest score was considered as the most redundant and could be pruned for further analysis.

ARTIFICIAL STRUCTURED DATA SETS

Three artificial structured data sets (ASDS) were used to compare pruning algorithms among themselves and with similar algorithms developed for fixed-size neural networks. Input x-values were generated by a random function, and the inputs were statistically independent for all data sets. The input and target values were scaled between 0.1 and 0.9 for network training.

The first two ASDS were produced analogously to ref 6 and had a general form

$$\text{target} = f(\theta) \equiv g(\theta) + \epsilon \tag{6}$$

where

$$\theta = \sum_{i=1}^{3} V_i x_i \tag{7}$$

$g(\theta)$ is an analyzed function, x_i are three independent variables, V_i are certain coefficients, and ϵ is a "noise" generated according to a normal distribution with a mean equal to zero. The x_i values were distributed over the interval [0,1], and the training sets contained 50 cases.

Linear $g(\theta)$. The function $g(\theta)$ in eq 6 was linear

$$g(\theta) = \theta \tag{8}$$

The coefficients V_1 and V_2 were constant values $V_i = \{-0.2, +0.4\}$, respectively; V_3 took one of the values $V_3 = \{0.01, 0.02, 0.05, 0.1, 0.2, 0.4\}$. Noise ϵ was added to keep the relationship between the independent variables and the target to a correlation coefficient of about 0.9.

Nonlinear $g(\theta)$. The target function $g(\theta)$ was generated by

$$g(\theta) = \begin{cases} [\theta - 0.5]^{1/2}, & \text{if } \theta > 0.5 \\ \sin(4\pi\theta), & \text{if } \theta \le 0.5 \end{cases} \tag{9}$$

Here V_i and x_i were the same as in the linear example. The

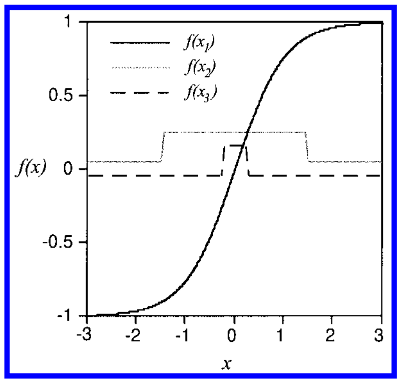

Figure 2. Model functions used in the additive function example.

correlation coefficient between $f(\theta)$ and $g(\theta)$ was 0.9. This was an example with a rather complex relationship between input variables and target function.

Additive Function. The third ASDS was given by the equation (Figure 2)

$$\text{target} = f(x) + \epsilon = f_1(x_1) + f_2(x_2) + f_3(x_3) + \epsilon \quad (10)$$

where

$$\begin{cases} f_1(x) = 10 \tanh(x) \\ f_2(x) = \tanh(100x + 150) - \tanh(100x - 150) \\ f_3(x) = \tanh(100x + 25) - \tanh(100x - 25) \end{cases} \quad (11)$$

The output is the sum of the three functions written above and some noise ϵ. Each of these three functions depends on only one input. The inputs were distributed over the interval $[-3, +3]$, and the training set contained 200 cases. A model like this, in which the output is the sum of univariate (nonlinear) transformations of the inputs, is called an "additive" model.[29]

The linear and nonlinear examples as well as the additive function allowed us to easily assess the importance of the inputs. The importance of input i in the first two ASDS is simply proportional to the magnitude of the corresponding coefficient V_i. The output generated by the additive function has two abrupt changes in response to x_2 and to x_3. The positions of these changes are farther from zero for x_2 than for x_3, so it is clear that for most practical purposes x_2 should be considered more important than x_3. It is also obvious that x_1 is associated with larger changes in the output than either x_2 and x_3, and it should be considered the most important input. It has been shown that this example could produce considerable difficulties for various sensitivity estimation methods, including optimal brain damage and some magnitude-based methods.[29] Two data sets with different levels of noise ϵ were generated. The noiseless example ($\epsilon = 0$) corresponded to that considered by Sarle.[29] In the second case, noise ϵ was added to keep the relationship between $f(x)$ and the target to a correlation coefficient of about 0.9; i.e., this example was more similar to real experimental data and to other examples considered in the article.

QSAR Data Sets. We applied the pruning methods to four sets of QSAR data. The first two data sets, 51 benzodiazepine derivatives with anti-pentylenetetrazole activity and 37 2,5-bis(1-aziridinyl)-*p*-carboquinones with antileukemic activity were previously investigated using MLR, functional-link-net (FUNCLINK),[30] and neural networks.[31] The next data studied were 74 2,4-diamino-5-(substituted benzyl)pyrimidines. The biological activity of pyrimidines was measured as the inhibition constant (log K_i) of dihydrofolate reductase from MB1428 *E. coli*.[32] These data were studied by several QSAR methods, including inductive logic programming and neural networks.[33] The three aforementioned QSAR examples were also analyzed by us using pruning methods developed for BNN.[6] The purpose of analysis of these data sets was to compare the performance of pruning algorithms. The last analyzed compounds were 53 antimycin analogues with antifilarial activity.[34] This set has often been studied in the QSAR literature using various statistical techniques and feature selection methods (see, e.g., ref 35 and references cited therein). It was used to test the prediction ability of the pruning algorithms using the LOO method and training/test set protocols.

CALCULATED RESULTS

Linear and Nonlinear ASDS. All analyzed pruning methods found input variable x_3 (it corresponds to the input neuron V_3) as redundant for $V_3 = \{0.01-0.1\}$. The order of sensitivities of the two other input variables—x_1 and x_2—corresponded to the magnitudes of the corresponding constants V_i. For example, the input variable x_2 was characterized by the largest magnitude of the constant $V_2 = 0.4$ in eq 7. The corresponding neuron was determined as the most important for network learning for all analyzed magnitudes of constant V_3, except when the magnitude of V_3 was equal to that of V_2 ($V_2 = V_3 = 0.4$). For the last case, both input neurons were equally important for network learning and had similar sensitivities (Table 1).

Pruning of x_3 resulted in some improvement of the predictive ability of neural networks for the range of values $V_3 = [0.01-0.05]$. The improvement was significant for the nonlinear example, but it practically did not influence results calculated for the linear example (Table 2). The influence of variable x_3 increased as a function of the magnitude of the constant V_3. This input variable eventually became more important than variable x_1 (characterized by constant $V_1 = 0.2$) for the largest value of $V_3 = 0.4$ as expected.

The overall results calculated by pruning methods developed for CCN coincided with those for BNN.[6] This is very striking considering the dramatic difference in architectures and training procedures of these types of neural networks.

Additive Function Analysis. The pruning methods applied for the noiseless data set ($\epsilon = 0$) found the input variable x_1 to be the most important and the input variable x_3 to be redundant; i.e., the calculated results are in complete agreement with theoretical expectations[29] and with results of consequent pruning of input variables (Table 3). Very similar sensitivities were calculated using BNN (data not shown). This result is different from that of Sarle,[29] who reported a failure of the same and some other similar sensitivity calculation methods to correctly estimate an order of importance of input neurons.

We note a difference in our approaches. Sarle[29] based his conclusion on theoretical analysis of one feed-forward artificial neural network with tanh(x) activation functions. The weights of his network were set to some analytical values

Table 1. Neural Network Sensitivities Calculated for Linear and Nonlinear Examples by Analyzed Pruning Methods

input variable	linear set			nonlinear set		
	A[a]	B	D	A	B	D
			$V_3 = 0.01$			
x_1	26[b]	0	8	25	39	27
x_2	2	0	0	5	3	5
x_3	72	100	92	70	58	68
			$V_3 = 0.02$			
x_1	24	0	9	20	35	33
x_2	0	0	4	3	0	6
x_3	76	100	87	77	65	61
			$V_3 = 0.05$			
x_1	21	0	10	21	27	24
x_2	2	0	2	4	2	6
x_3	77	100	88	75	71	70
			$V_3 = 0.1$			
x_1	15	1	7	33	39	34
x_2	1	0	3	5	4	3
x_3	84	99	90	62	57	63
			$V_3 = 0.2$			
x_1	25	4	11	54	59	46
x_2	2	0	3	5	5	18
x_3	73	96	86	41	36	36
			$V_3 = 0.4$			
x_1	59	65	68	80	88	75
x_2	2	23	8	7	7	14
x_3	39	12	24	13	5	11

[a] Sensitivity calculation method. [b] The number of cascade-correlation neural networks (100 networks were used in ensemble), when the input variable shown in the first row had the lowest sensitivity. The results are shown at an early stopping point. Note, the higher scores correspond to the least important input variables.

Table 2. Leave-One-Out Correlation Coefficients Calculated by Cascade-Correlation Neural Network for Linear and Nonlinear Examples with and without (in Parentheses) Input Variable x_3

magnitude of constant V_3	linear	nonlinear
0.01	0.880 ± 0.002 (0.883 ± 0.004)[a]	0.78 ± 0.01 *(0.85 ± 0.01)*
0.02	0.882 ± 0.004 (0.882 ± 0.004)	0.79 ± 0.01 *(0.84 ± 0.01)*
0.05	0.884 ± 0.004 (0.880 ± 0.004)	0.83 ± 0.01 *(0.85 ± 0.01)*
0.1	0.880 ± 0.002 (0.861 ± 0.004)	0.86 ± 0.02 (0.82 ± 0.01)
0.2	0.883 ± 0.004 (0.816 ± 0.002)	0.78 ± 0.01 (0.64 ± 0.02)
0.4	0.887 ± 0.002 (0.670 ± 0.010)	0.73 ± 0.03 (0.59 ± 0.02)

[a] The limits at confidence level $p < 0.05$ are indicated. The limits correspond to two standard deviations from mean value of the corresponding parameters calculated for 10 ensembles of cascade-correlation neural networks (each ensemble was formed by 100 neural networks). The same method to estimate limits was used in all other reported tables. In italic are shown sets where an improvement of prediction ability was observed after pruning of the input variable x_3.

in such a way that the output *exactly* formed the target function for analysis. There was not any training procedure of the neural network.

Figure 3 demonstrates the dynamics of change of the input neuron sensitivities with duration of the CCN training for method B. It is possible to see that the largest difference in sensitivities of input neuron x_1 compared to that of x_2 and x_3 is for CCN with only a few hidden neurons. We suppose that this corresponds to the fact that during this time CCN learned the main dependency between the output and input neuron x_1. The difference decreased with the duration of

Table 3. Sensitivities and Statistical Coefficients Calculated for the Additive Function

pruned variable	method			leave-one-out estimations	
	A	B	D	correlation coeff, R	cross-validated, q^2
			Noiseless Data Set		
				0.996 ± 0.001[a]	0.991 ± 0.002[b]
x_1	6	0	17	-0.3 ± 0.1	-0.02 ± 0.06
x_2	10	19	12	0.990 ± 0.01	0.980 ± 0.001
x_3	84	81	71	0.995 ± 0.001	0.990 ± 0.002
			Data Set with Added Noise		
				0.883 ± 0.002	0.780 ± 0.006
x_1	9	0	5	-0.61 ± 0.2	-0.02 ± 0.02
x_2	43	46	47	0.887 ± 0.002	0.787 ± 0.004
x_3	48	54	48	0.886 ± 0.004	0.784 ± 0.006

[a] Coefficients calculated when using all three input variables. [b] Statistical coefficients computed without the input variable shown in the first column.

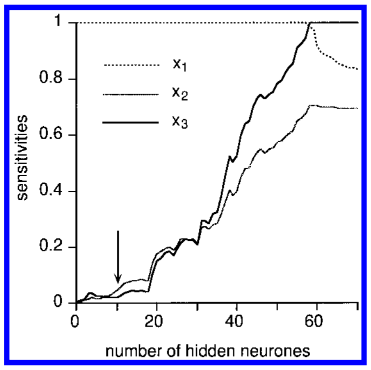

Figure 3. Sensitivities of input neurons calculated by method B for the additive function as a function of the training time of the cascade-correlation neural network. All sensitivities were normalized to a maximum value of 1. The early stopping point (shown by arrow) was detected for a network with 10 hidden neurons. The neural network calculated the correct order of sensitivities (i.e., $x_1 > x_2 > x_3$) for this point. Further training of the algorithm results in the incorrect order of sensitivities due to the overfitting/overtraining problem.

training; i.e., the significance of two other input neurons increased. This change corresponded to a shift of attention of the artificial neural network, from learning of the main dependency within the data set to learning of particular small dependencies due to the presence of inputs x_2 and x_3. The number of used data points in learning data sets (100) was enough for confident learning of such dependencies (as it is monitored by the prediction ability of the network for the validation data set) up to the early stopping point (shown by arrow), i.e., 10 neurons in the hidden layer. After the early stopping point the number of used data was not adequate to provide an improvement of accuracy for the validation data set; i.e., the neural network started to overfit the data. The overfitting eventually led to an incorrect ordering of sensitivities of neurons. This example showed that the process of neural network training is an important factor for correct calculation of input neuron sensitivities.

The presence of noise in training data sets affected the order of importance of the inputs for neural network regression. The pruning of each of two inputs x_2 or x_3 provided similar prediction ability, which was slightly improved compared to an analysis using the total set of inputs (Table 3). This change in importance of inputs was properly tracked by the pruning methods. The sensitivities calculated

Table 4. Leave-One-Out Results for QSAR Examples

(a) 51 Benzodiazepine Derivatives

sensitivity method	analyzed input variables							correlation coeff, R	cross-validated q^2
	MR-3	PI-3	MR-7	σ_m-3	F-4	R-4	I-7		
A	2[a]	*42*	2	4	3	38	9		
B	0	*46*	2	1	0	41	10	0.80 ± 0.01[b] (0.80)	0.64 ± 0.02[b] (0.64 ± 0.03)
D	4	*45*	2	1	4	33	11		
A	4	×[c]	2	5	4	*67*	18		
B	2	×	2	1	1	*71*	23	0.81 ± 0.01 (0.81)	0.65 ± 0.02 (0.66 ± 0.03)
D	9	×	4	1	7	*59*	20		
		×				×		0.82 ± 0.01 (0.82)	0.66 ± 0.01 (0.67 ± 0.02)

(b) 37 Carboquinone Derivatives

sensitivity method	analyzed input variables						correlation coeff, R	cross-validated q^2
	$MR_{1,2}$	$PI_{1,2}$	PI_2	MR_1	$\boldsymbol{F}$	$\boldsymbol{R}$		
A	18	6	25	7	8	36		
B	*34*	4	32	20	4	6	0.87 ± 0.01 (0.89)	0.76 ± 0.02 (0.79 ± 0.03)
D	22	6	*35*	13	20	4		
A	×	6	*44*	7	9	34		
B	×	6	*66*	19	6	3	0.87 ± 0.01 (0.92)	0.76 ± 0.02 (0.84 ± 0.02)
D	×	4	*56*	15	22	3		
	×		×				0.88 ± 0.01 (0.93)	0.78 ± 0.02 (0.85 ± 0.02)

[a] The number of cascade-correlation neural networks (100 networks were used in ensemble), when the input variable shown in the first row had the lowest sensitivity. The results are shown at an early stopping point. Note, the higher scores (in italics) correspond to the least important input variables. [b] The values computed by cascade-correlation and fixed-size back-propagation neural networks (in parentheses). The limits are calculated at confidence level $p < 0.05$ (see footnotes of Table 2 for details of calculation). [c] × denotes the variable that was not used in the analysis.

Table 5. Results for Sets of Variables Determined by Pruning Methods[a]

(a) Pyrimidines

method	selected variables	correlation coeff, R		cross-validated q^2	
		CCN	BNN[a]	CCN	BNN
	all variables, 1−27	0.62 ± 0.03	0.63 ± 0.02	0.37 ± 0.05	0.40 ± 0.03
A	2, 3, 4, 6, 8, 11, 12, 17, 20, 23	0.78 ± 0.02	0.79 ± 0.02	0.62 ± 0.02	0.63 ± 0.03
B	2, 3, 11, 12, 20, 23	0.83 ± 0.01	0.81 ± 0.01	0.68 ± 0.01	0.65 ± 0.02
D	2, 3, 4, 8, 11, 12, 15, 20, 23	0.82 ± 0.01	0.80 ± 0.01	0.68 ± 0.01	0.64 ± 0.02

(b) Antifilarial Antimycin Analogues

method	selected variables	correlation coeff, R			cross-validated q^2		
		MLR[b]	CCN	BNN[c]	MLR	CCN	BNN
	all variables, 1−53		0.57 ± 0.03	0.57 ± 0.03		0.32 ± 0.03	0.32 ± 0.03
	24, 50, 51[d]	0.65	0.68 ± 0.02	0.66 ± 0.02	0.39	0.46 ± 0.02	0.43 ± 0.03
A	11, 13, 14, 35, 50, 52	0.80	0.80 ± 0.02	0.82 ± 0.01	0.62	0.64 ± 0.02	0.67 ± 0.01
B	4, 6, 11, 13, 14, 35, 50, 52	0.82	0.82 ± 0.01	0.82 ± 0.01	0.66	0.67 ± 0.01	0.67 ± 0.01
D	4, 6, 7, 11, 13, 14, 35, 50, 52	0.82	0.81 ± 0.01	0.81 ± 0.01	0.64	0.66 ± 0.02	0.66 ± 0.02

[a] The results were calculated using the leave-one-out method. [b] MLR = multiple linear regression; BNN = fixed-size back-propagation neural networks; CCN = cascade-correlation neural network. [c] BNN had one hidden layer with 10 neurons and were trained as described in ref 6. [d] Variables selected by the ARTHUR program.[34]

for input variables x_2 and x_3 by all methods were practically equal.

Analysis of QSAR Data Sets. The results calculated by CCN for the first two data sets (Table 4) were similar to those reported for BNN.[6] All pruning methods excluded variables PI-3 and R-4 for the benzodiazepine derivatives. The methods B and D excluded variables $MR_{1,2}$ and PI_2 for carboquinones. The same variables were excluded by pruning methods developed for fixed-size neural networks[6] and by FUNCLINK.[31] The prediction ability of neural networks—measured by the LOO procedure—did not change after pruning of the redundant inputs. For this data set the correlation coefficient R and cross-validated q^2 were lower for CCN compared to values calculated using BNN (Table 4b). Method A failed to find a best set of variables for the carboquinones. This method determined variable $\boldsymbol{R}$ as redundant during the first step of the pruning procedure. Elimination of this variable decreased the prediction ability of the neural networks to $R = 0.82 \pm 0.02$ and $q^2 = 0.67 \pm 0.02$ which was lower than the results calculated using other methods.

Application of the pruning algorithms to the analysis of the pyrimidines substantially improved the predictive ability of neural networks (see Table 5a). All methods found very similar sets of parameters. The variables determined by method B represented a “kernel set” that was included in the best sets of methods A and D. Four and three additional variables were presented in the sets revealed by pruning methods A and D, respectively. A detailed examination of sets found by consequent pruning showed that the set determined by method B is minimal. The predictive ability of neural networks significantly decreased after pruning of

Table 6. Statistical Coefficients Calculated for Antifilarial Antimycin Analogues Using Training/Test Sets Protocols

		orrelation coeff, R		cross-validated q^2	
method	selected variables	LOO,[a] validation set	test set	LOO, validation set	test set
		Training Set Composed of Molecules 1−16			
	all variables, 1−53	0.42 ± 0.05	0.61 ± 0.03	0.2 ± 0.1	0.35 ± 0.03
A	4, 6, 13, 14, 35, 46, 50, 51	0.87 ± 0.02	0.65 ± 0.02	0.72 ± 0.02	0.37 ± 0.03
B	4, 6, 50, 51	0.87 ± 0.02	0.75 ± 0.01	0.74 ± 0.02	0.51 ± 0.02
D	4, 6, 13, 14, 46, 50, 51	0.88 ± 0.02	0.68 ± 0.03	0.72 ± 0.02	0.42 ± 0.02
		Training Set Composed of Molecules 17−31			
	all variables, 1−53	0.48 ± 0.04	0.55 ± 0.03	0.22 ± 0.1	0.26 ± 0.08
A, B, D	29, 33, 50	0.73 ± 0.02	0.56 ± 0.03	0.52 ± 0.04	0.32 ± 0.04

any parameter from the kernel set, while all additional variables could be pruned for sets detected by methods A and D. This analysis showed that sometimes sets of parameters determined by pruning methods could still contain some redundant variables. The presence of such parameters does not improve the prediction ability of neural networks compared to the kernel set. An application of consequent pruning could further refine the detected sets of variables. However, this method represents a rather slow computational algorithm and can be used only for final analysis of small sets of compounds.

The prediction abilities of CCN and BNN for the same sets of parameters were very similar. It is interesting to note that pruning methods developed for the cascade-correlation learning architecture detected sets of variables that were characterized by better prediction ability compared to sets selected using the same methods for BNN,[6] where the best set was found by pruning method E (this method was not used in the current study) and was characterized by correlation coefficient $R = 0.78$ and cross-validated $q^2 = 0.61 \pm 0.04$.

The last analyzed QSAR set of antifilarial antimycin analogues was composed of 31 molecules and 53 descriptors. Neural networks applied to the set of variables 24, 50, and 51 selected by the ARTHUR program[34] showed better prediction ability compared to analysis of the total set of descriptors (Table 5b). The statistical coefficients calculated by the LOO method were similar for artificial neural networks and linear regression. An application of pruning algorithms detected sets of variables with higher prediction ability compared to the set detected by multivariate analysis. The best sets of parameters were determined by pruning methods B and D.

In the present study selection of parameters was done using all available molecules in the initial training data set (e.g., 31 for antimycin analogues). Actually, only 50% of molecules was available for neural network learning while the remaining 50% of molecules was used to determine the early stopping point and composed the validation set. The selected variables were thus rigorously validated, and we could expect that such methods provide a satisfactory prediction accuracy of selected sets of variables for analysis of new molecules.

To verify this assumption and to further validate the prediction ability of the elaborated methods, we used the training/test set protocol as indicated in the original study of Selwood et al.[34] The first 16 molecules were chosen as the training set, while the remaining 15 molecules formed the test set and never participated in neural network learning. The selected variables (Table 6) contained several descriptors, e.g., 4, 6, 50, and 51, that were found to be significant when analyzing the whole set of molecules.

Table 7. Sets of Variables Determined for Antifilarial Antimycin Analogues Using Several Runs of Pruning Algorithm

method	set of variables	correlation coeff, R	cross-validated q^2
B	4, 6, 11, 14, 35, 50, 52	0.81 ± 0.02	0.65 ± 0.02
	4, 6, 11 , 13, 14, 35, 50, 52	0.82 ± 0.02	0.67 ± 0.02
	4, 6, 11, 14, 50, 52	0.77 ± 0.02	0.59 ± 0.02
	4, 6, 11, 13, 14, 35, 50, 52	0.82 ± 0.02	0.67 ± 0.02
	4, 6, 14, 50	0.78 ± 0.02	0.60 ± 0.02
D	4, 6, 7, 11, 13, 14, 35, 50 52	0.81 ± 0.02	0.66 ± 0.02
	4, 6, 14, 50	0.77 ± 0.02	0.60 ± 0.02
	4, 6, 11, 14, 45, 50, 52	0.76 ± 0.02	0.57 ± 0.03
	2, 4, 6, 11, 14, 41, 50, 52	0.78 ± 0.02	0.61 ± 0.02
	2, 4, 6, 11, 14, 41, 50, 52	0.79 ± 0.02	0.61 ± 0.02

In the next stage of the analysis the training and test set of molecules were reversed. The best sets of variables detected by all pruning methods coincided and were composed of three descriptors, 29, 33, and 50. These results demonstrated that variable selection critically depends on the composition of sets used to train and to check the performance of artificial neural networks.

The last method for the validation of pruning algorithms was full cross-validation LOO.[35] In this method each of the 31 molecules was removed one at a time, both from the training and the validation data sets, and the best models were detected for the reduced sets of the remaining 30 molecules. The selected variables were used to predict activity of only one molecule, i.e., the molecule that did not participate for the variable selection procedure at that step. Since such analysis required a lot of calculations, only pruning method B was investigated. The statistical parameters found in this way, $R = 0.65 \pm 0.02$ ($q^2 = 0.38 \pm 0.02$), were lower compared to the results reported in Table 5. However, the prediction ability of the pruned set of parameters was still significantly higher compared to analysis without optimization of parameters (Table 5).

The pruning algorithms showed some variations when determining a best set of parameters due to different initializations of neural network weights, as is shown for variables selected by pruning methods B and D for the antimycin analogues (Table 7). Despite some variations of the number of parameters in sets selected by a pruning method, it was always possible to determine some kernel set that was present in almost all found sets. For example, variables determined by pruning methods B and D always included parameters 4, 6, 14, and 50. We note that the set of parameters—11, 13, 14, 35, 50, 52—detected by method A did not include variables 4 and 6, but it included variable 52. This variable had a correlation coefficient $R = -0.71$

with variable 6, and it compensated for the absence of variable 6. Probably, a combination of other variables presented in the set also compensated for the absence of the variable 4.

DISCUSSION

This study introduced pruning methods able to find best sets of descriptors for CCN. The results calculated by these methods are comparable to those previously reported for methods elaborated for fixed-size neural networks. The prediction ability of the CCN and BNN for the same sets of variables was very similar on average. However, for some data (e.g., carboquinone derivatives) the prediction ability of CCN was lower, while for others (e.g., variables selected by methods B and D for pyrimidines) it was higher compared to BNN. This indicates a dependence of relative performance of both types of neural networks on the internal structure of the analyzed data. It is interesting to note that in our analysis of QSAR data we did not find examples where the performance of CCN methods was dramatically better compared to that of BNN, such as for the two-spirals problem[17] reported elsewhere. This might be due to the relatively simple dependencies between the input variables and investigated activities of the analyzed data sets. We cannot discard the possibility that for different data sets CCN could significantly overperform BNN. Thus, the CCN pruning methods proposed in this study could be offered as the only suitable ones to be used in such studies. The present state of the art of artificial neural network theory cannot answer unambiguously a question about which type of neural networks (i.e., topology-modifying or fixed-size neural networks) is better for data modeling. This question should be carefully addressed by experimental studies. A dramatic increase in the power of new computers makes it possible to apply both types of artificial neural networks simultaneously.

We emphasize the importance of using an ensemble of neural networks to find the best set of variables and to avoid the problem of chance correlation. An analysis of sensitivities of input neurons in any single neural network can be biased, and even the most important variables could be detected as nonsignificant and pruned. For example, the results presented in Table 3 show that method D determined input variable x_3 as redundant in 17 out of 100 analyzed networks. An analysis of any of these 17 networks could provide incorrect results. However, the analysis of an ensemble of 100 networks correctly determined variable x_1 as redundant.

The results show that pruning methods may select different sets of variables even if they are applied to the same data. The origin of the variation in selection of variables is 2-fold. First, the proposed methods calculate similar sensitivities for correlated parameters. That is why it is possible that at some level of pruning the parameter selection procedure will go by different ways, depending on which one of two correlated parameters was pruned. This can explain the variability of parameters selected by the same, or different pruning, methods. Second, because of variations of neural network prediction (note that pruning is stopped according to the minimal value of RMSE and this value is always determined with some final precision), it is possible that redundant descriptors could still remain in the final set of selected variables. The prediction ability of such sets with redundant variables is the same as that of the set without these variables. The possibility that one or both of these problems will take place critically depends on the internal structure of the analyzed data. We compare the distribution of sensitivities for benzodiazepine and carboquinone derivatives. At the first step of analysis of benzodiazepines, more than 80% of neural networks from an ensemble correctly find parameters PI-3 and R-4 as redundant. The internal structure of these data is such that pruning methods easily evaluate the importance of parameters. As to carboquinone derivatives it is obvious that on average three parameters (e.g., for method A these parameters were $MR_{1,2}$, PI_2, $\boldsymbol{R}$) have similar high scores for each pruning method. These data have a more complex internal structure for pruning. One possible explanation is that some input variables are highly correlated, e.g., the correlation coefficient for parameters $PI_{1,2}$ and PI_2 is 0.9, for $MR_{1,2}$ and MR_1 is 0.67, and for $\boldsymbol{F}$ and $\boldsymbol{R}$ is about 0.66. Probably, this fact prevents method A from determining the correct order of variable pruning. One can easily resolve this problem by using consequent pruning of all variables having high redundancy scores. However, such analysis can be recommended only for data with a small amount of variables since it is very time-consuming.

The presence of outliers in the feature variables can also result in the instability of the chosen variables. We suppose that the presence of outliers in the antimycin analogues[35] can be one of the reasons why results calculated using the first 16 compounds in the training set are significantly higher compared to those calculated using the remaining molecules (Table 5b). Indeed, outliers have already been pointed out in this set.[36] This supposition requires further studies.

It is interesting to compare the performance of parameters determined by pruning methods with those determined by Dunn and Rogers.[35] The best correlation coefficient calculated using a parameter selection procedure based on the analysis of the complete data set is the same $R = 0.82$ for CCN and for genetic function approximation (GFA) methods. This result indicates similar performance of both algorithms and, probably, absence of significant nonlinear dependencies between parameters and activities of the molecules. It should be noted that in order to improve the performance of their methods an outlier limiting procedure was used by Dunn and Rogers,[35] while any similar procedure was not implemented in the current study. The identical performance of both algorithms suggests that outlier limiting is done to some extent by CCN. This fact is easy to understand by taking into account the nonlinear properties of the logistic activation function $f(x) = 1/(1 + e^{-x})$ of neural network neurons. However, it is possible that use of an advanced outlier limiting procedure could further improve the calculated results. The correlation coefficient determined for antimycin derivatives using the full cross-validation procedure is higher for CCN $R = 0.65 \pm 0.02$ compared to that of GFA $R = 0.57$, and it is in the top range of values (0.55−0.66) determined using the genetic partial least squares (GPLS) algorithm. However, 30 variables were required for optimal performance of GPLS, while only a few parameters are required for similar performance of neural networks. This is an important advantage of CCN over GPLS since more easy interpretation of calculated results (at least quantitative) can be done for artificial neural networks. It is interesting to note that the parameters selected by CCN also provide high prediction ability for MLR. These results suggest to us that use of an ensemble of neural networks trained with only 50% of the available molecules makes it possible to do a robust optimization of input variables.

In summary, we show that the elaborated pruning methods can be successfully used to optimize the set of variables for neural networks trained with the cascade-correlation learning algorithm. Despite the variability of selected variables the pruning algorithms always tend to determine some kernel set of descriptors that are probably the most relevant to the analyzed activities. However, the composition of training sets and, more particularly, the presence of outliers could influence the variable selection procedure, as was demonstrated for the Selwood data set. The use of variables selected by pruning methods could provide an improvement of neural network prediction ability compared to that calculated using the unpruned sets of variables.

ACKNOWLEDGMENT

This study was partially supported by INTAS-Ukraine 95-0060 and the Swiss National Science Foundation FNRS 2150-045689.95 grants. The authors thank Brian Hyland for his helpful comments.

REFERENCES AND NOTES

(1) Devillers, J., Ed. *Neural Networks in QSAR and Drug Design*; Academic Press: London, 1996.

(2) Maddalena, D. Applications of Artificial Neural Networks to Quantitative Structure−Activity Relationships. *Exp. Opin. Ther. Patents* **1996**, *6*, 239−251.

(3) Zupan, J.; Gasteiger, J. *Neural Networks for Chemists: An Introduction*; VCH Publisher: Weinheim, Germany, 1993.

(4) Aivazyan, S. A.; Buchstaber, V. M.; Yenyukov, I. S.; Meshalkin, L. D. *Applied Statistics. Classification and Reduction of Dimensionality*; Finansy i statistika: Moscow, 1989.

(5) Tetko, I. V.; Kovalishin, V. V.; Luik, A. I. Neural Network Pruning Algorithm That Estimates the Input Parameter Performances. *Dopov. Akad. Nauk Ukr. (Proc. Ukrainian Acad. Sci.)* **1995**, *7*, 63−66.

(6) Tetko, I. V.; Villa, A. E. P.; Livingstone, D. J. Neural Network Studies. 2. Variable Selection. *J. Chem. Inf. Comput. Sci.* **1996**, *36*, 794−803.

(7) Maddalena, D. J.; Johnston, J. A. R. Prediction of Receptor Properties and Binding Affinity of Ligands to Benzodiazepines/GABAA Receptors Using Artificial Neural Networks. *J. Med. Chem.* **1995**, *38*, 715−724.

(8) Wikel, J. H.; Dow, E. R. The Use of Neural Networks for Variable Selection in QSAR. *Bioorg. Med. Chem. Lett.* **1993**, *3*, 645−651.

(9) Tetko, I. V.; Villa, A. E. P.; Aksenova, T. I.; Zielinski, W. L.; Brower, J.; Collantes, E. R.; Welsh, W. J Application of a Pruning Algorithm To Optimize Artificial Neural Networks for Pharmaceutical Fingerprinting. *J. Chem. Inf. Comput. Sci.* **1998**, *38*, 676−684.

(10) Tetko, I. V.; Tanchuk, V. Yu.; Chentsova, N. P.; Antonenko, S. V.; Poda, G. I.; Kukhar, V. P.; Luik, A. I. HIV-1 Reverse Transcriptase Inhibitor Design Using Artificial Neural Networks. *J. Med. Chem.* **1994**, *37*, 2520−2526.

(11) Livingstone, D. J.; Manallack, D. T.; Tetko, I. V. Data Modeling with Neural Networks: Advantages and Limitations. *J. Comput.-Aided Mol. Des.* **1997**, *11*, 135−142.

(12) Hosseini, M.; Maddalena, D. J.; Spence, I. Using Artificial Neural Networks to Classify the Activity of Capsaicin and Its Analogues. *J. Chem. Inf. Comput. Sci.* **1997**, *37*, 1129−1137.

(13) Reed, R.; Marks, R. J., II. Neurosmithing: Improving Neural Network Learning. In *The Handbook of Brain Theory and Neural Networks*; Arbib, M. A., Ed.; MIT Press: Cambridge, MA, London, 1995; pp 639−644.

(14) Geman, S.; Bienenstock, E.; Dourstat, R. Neural Networks and the Bias/Variance Dilemma. *Neural Comput.* **1992**, *4*, 1−58.

(15) Tetko, I. V.; Livingstone, D. J.; Luik, A. I. Neural Network Studies. 1. Comparison of Overfitting and Overtraining. *J. Chem. Inf. Comput. Sci.* **1995**, *35*, 826−833.

(16) Ash, T.; Cottrell, G. Topology-Modifying Neural Network Algorithms. In *The Handbook of Brain Theory and Neural Networks*; Arbib, M. A., Ed.; MIT Press: Cambridge, MA, London, 1995; pp 990−993.

(17) Fahlman, S. E.; Lebiere, C. The Cascade-Correlation Learning Architecture. In *NIPS*2*; Touretzky, D. S., Ed.; Morgan-Kaufmann: San Mateo, CA, 1990; pp 524−532.

(18) Tetko, I. V.; Villa, A. E. P. An Enhancement of Generalization Ability in Cascade Correlation Algorithm by Avoidance of Overfitting/Overtraining Problem. *Neural Process. Lett.* **1997**, *6*, 43−50.

(19) Tetko, I. V.; Villa, A. E. P. Efficient Partition of Learning Data Sets for Neural Network Training. *Neural Networks* **1997**, *10*, 1361−1374.

(20) In the early version of the ESE algorithm we used the term "control set" instead of "validation set". The term validation set is generally accepted in neural network literature (see for example: Leblanc, M.; Tibshirani, R. Combining Estimates in Regression and Classification. *J. Am. Statist. Assoc.* **1996**, *91*, 1641−1650), and we recommend using this term instead of control set.

(21) Cramer, R. D., III; DePriest, S. A.; Patterson, D. E.; Hecht, P. The Developing Practice of Comparative Field Analysis. In *3D QSAR in Drug Design: Theory Methods and Applications*; Kubinyi, H., Ed.; ESCOM: Leiden, The Netherlands, 1993; pp 443−486.

(22) Fahlman, S. E. Faster-Learning Variations on Back-Propagation: An Empirical Study. *Proceedings of the 1988 Connectionist Models Summer School*; Morgan Kaufmann: San Mateo, CA, 1988.

(23) LeCun, Y.; Dencer, J. S.; Solla, S. A. Optimal Brain Damage. In *NIPS*2*; Touretzky, D. S., Ed.; Morgan Kaufmann: San Mateo, CA, 1990; pp 598−5605.

(24) Hassibi, B.; Stork, D. Second-Order Derivatives for Network Pruning: Optimal Brain Surgeon. In *NIPS*5*; Hanson, S., Cowan, J., Giles, C., Eds.; Morgan-Kaufmann: San Mateo, CA, 1993; pp 164−171.

(25) Hansen, L. K.; Rasmussen, C. E. Pruning from Adaptive Regularization. *Neural Comput.* **1994**, *6*, 1223−1233.

(26) Weigen, S. A.; Rumelhart, D. E.; Huberman, B. A. Generalization by Weight-Elimination with Application to Forecasting. In *NIPS*3*; Lippann, R., Moody, J., Touretzky, D. S., Eds.; Morgan Kaufmann: San Mateo, CA, 1991; pp 875−882.

(27) Chauvin, Y. A. Back-Propagation Algorithm with Optimal Use of Hidden Units. In *NIPS*1*; Touretzky, D. S., Ed.; Morgan Kaufmann: San Mateo, CA, 1989; pp 519−526.

(28) Reed, R. Pruning Algorithms−A Survey. *IEEE Trans. Neural Networks* **1993**, *4*, 740−747.

(29) Sarle, W. S. *How To Measure the Importance of Inputs?* SAS Institute Inc.: Cary, NC, 1996 (available in electronic form as ftp://ftp.sas.com/pub/neural/importance.html).

(30) Nakao, H.; Arakawa, M.; Nakamura, T.; Fukuchima, M. Antileuketic Agents. II. New 2,5-bis(1-aziridinyl)-p-benzoquinone Derivatives. *Chem. Pharm. Bull.* **1972**, *20*, 1968−1979.

(31) Liu, Q.; Hirono, S.; Moriguchi, I. Comparison of Functional-Link Net and the Generalized Delta Rule Net in Quantitative Structure−Activity Relationship Studies. *Chem. Pharm. Bull.* **1992**, *40*, 2962−2969.

(32) Li, R. L.; Hansch, C.; Kaufman, B. T. A Comparison of the Inhibitory Action of 5-(Substituted benzyl)-2,4-diaminopyrimidines on Dihydrofolate Reductase from Chicken Liver with That from Bovine Liver. *J. Med. Chem.* **1982**, *25*, 435−440.

(33) Hirst, J. D.; King, R. D.; Sternberg, M. J. E. Quantitative Structure−Activity Relationships by Neural Networks and Inductive Logic Programming. 1. The Inhibition of Dihydrofolate Reductase by Pyrimidines. *J. Comput.-Aided Mol. Des.* **1994**, *8*, 405−420.

(34) Selwood, D. L.; Livingstone, D. J.; Comley, J. C. W.; O'Dowd, A. B.; Hudson, A. T..; Jackson, P.; Jandu, K. S.; Rose, V. S.; Stables, J. N. Structure−Activity Relationships of Antifilarial Antimycin Analogues: A Multivariate Pattern Recognition Study. *J. Med. Chem.* **1990**, *33*, 136−142.

(35) Dunn, W. J.; Rogers, D. Genetic Partial Least Squares in QSAR. In *Genetic Algorithms in Molecular Modeling*; Devillers, J., Ed.; Academic Press: London 1996; pp 109−130.

(36) Livingstone, D. J. The Trouble with Chemometrics. In *QSAR and Molecular Modeling: Concepts, Computational Tools and Biological Applications*; Sanz, F., Giraldo, J. , Manaut, F., Eds.; J. R. Prous: Barcelona, 1995; pp 18−26.

CI980325N