Computational Neural Networks as Model-Free Mapping Devices

G. M. Maggiora* and D. W. Elrod*

Upjohn Laboratories, 301 Henrietta Street, Kalamazoo, Michigan 49001

R. G. Trenary

Department of Computer Science, Western Michigan University, Kalamazoo, Michigan 49008

Received June 29, 1992

Computational neural networks (CNNs) represent a set of computational paradigms with the power to address a wide range of problems from pattern recognition to system identification. The present work focuses on a number of issues that arise in essentially all CNN applications and are especially important in chemical applications of quantitative structure–activity and structure–property relationships (QSAR and QSPR, respectively). Issues related to network optimization, data representation, error analysis, and generalization are identified, and their significance to CNN applications is described. A three-dimensional response surface designed to model many problems associated with QSAR and QSPR predictions is described, and the results of extensive CNN experiments based on this surface explicitly address a number of the issues. Special emphasis is placed on the critical issues of small data sets and noisy data that plague many chemical applications of neural nets.

INTRODUCTION

Although based loosely on analogies to the brain, neural networks derive their power not from their ability to model the brain, but rather from their ability to treat a vast array of diverse problems using a highly-distributed parallel computer architecture.[1-6] Neural networks, sometimes called artificial neural networks or computational neural networks (CNNs) to explicitly indicate their function as computational devices and not as brain models, are made up of collections of highly-interconnected but relatively simple processing elements (PEs); each interconnection has an associated weight that specifies the strength of the connection. In general, CNNs are distinguished by their network paradigms, which define the nature of their PEs, the network topology or pattern of connectivity among the PEs, and the learning method. There are two categories of learning methods, unsupervised and supervised.[1-5] In unsupervised learning the network itself determines the appropriate set of weights, while in supervised learning weights are determined such that the error between a set of training data and network predictions is minimized.[7] Unlike traditional computers that store programs and data separately, CNNs store information in the distributed pattern of their interconnections and values of the associated weights.

Although many CNN paradigms have been investigated, we will focus our attention here on multilayer, feedforward nets, called generalized perceptrons,[8] with either back-propagation or stochastic supervised learning.[9,10] These CNNs have been successfully applied to a wide variety of problems and are the most extensively studied to date. A typical example of such a CNN is depicted in Figure 1. As shown in Figure 1a, each PE performs two operations, a summation denoted by "Σ", followed by a function evaluation denoted by "f"; f is called a transfer function[11] and is generally taken to be either a sigmoid or the closely related hyperbolic tangent function, *tanh*, shown in Figure 1b.[12] Each PE also possesses a threshold or "bias weight", generally denoted Θ_i, which when combined with a constant, unit signal, effectively shifts the transfer function along its abscissa (cf. Figure 1).

CNNs can be viewed in two ways, either as classifying or as mapping devices. Generally, but not always, classification represents a less stringent, although nontrivial, test of a CNN's performance than function mapping. This follows from the fact that in the latter case the value of the function over the domain of interest must be predicted, while in the former case only whether the function value lies above or below some threshold is required. Nevertheless, it has been shown by numerous workers that generalized perceptrons can accurately represent the mappings of essentially all reasonably well-behaved functions.[13-17] The proofs, however, are not constructive and, thus, do not indicate precisely how one should produce a CNN that can, in fact, carry out the desired mapping.

Kosko[18] has pointed out that CNNs can be considered as *model-free* mapping devices in so far as the functional form of the mapping need not be specified explicitly, in contrast to the situation in both linear and nonlinear regression methods.[19] This would appear to provide an advantage to CNNs in cases where complicated input–output relationships are an inherent feature of the system or systems under investigation.[20] To investigate such systems, however, is difficult due to the problem of finding suitably complete data sets. Thus, the present work focuses on the study of a well-defined model system designed to simulate the salient features of response surfaces or functions, R, such as would typically be encountered in structure–activity or structure–property studies (vide infra). A distinct advantage of this approach is that a number of important problems can be addressed in a more "controlled" manner.

IMPORTANT COMPUTATIONAL NEURAL NETWORK ISSUES

Network Optimization. Network optimization represents an important aspect of CNNs that must be addressed and includes considerations of the optimal number of PEs or nodes, the nature of the error function,[21] and the nature of the transfer function. Once a particular CNN paradigm is chosen—generalized perceptrons in the present work—one of the most difficult challenges to the development of CNN applications is the determination of an appropriate network topology, i.e., the number of nodes and interconnections. As each node has

0095-2338/92/1632-0732$03.00/0

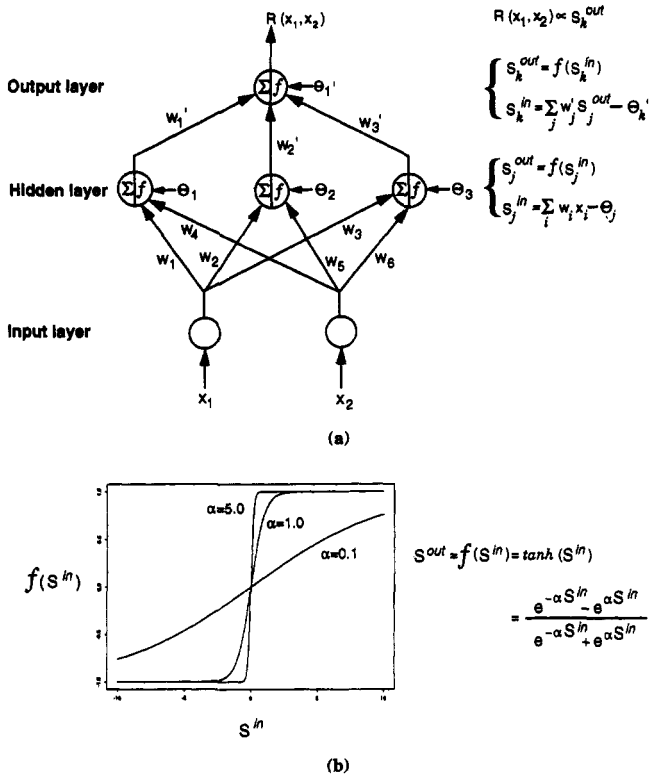


Figure 1. (a) Schematic depiction of a typical three-layer, feedforward computational neural network (CNN). Each layer of the net consists of a set of nodes (indicated by the circles), joined by weighted, unidirectional connections (indicated by the arrows). The values of the two input variables, x_1 and x_2, are passed through the input nodes without change. The remaining nodes are called processing elements (PE) and carry out both a summation of the incoming signals (denoted by Σ) and an evaluation of the resulting summand by a nonlinear transfer function, denoted by f. The detailed form of these functions are shown to the right. In addition to the weighted inputs summed by each PE, an additional threshold or bias term (denoted by Θ_i) is also added to the sum. The final output, $R(x_1,x_2)$, is then obtained by an appropriate scaling of the value of the transfer function, S_k^{out}, in the output layer. Scaling is required as the output range of most transfer functions are given by (0,1) or (−1,1), as seen for the *tanh* transfer function depicted in Figure 1b. (b) Example of the commonly used *tanh* transfer function. Note that the form of the function is quite sensitive to the gain parameter, α, which can change the function from one that resembles a step function to one that approximates a straight line. A value of $\alpha = 1.0$ is used in the present study. Although not depicted explicitly here, the threshold or bias term, Θ_i, shifts the transfer function along the abscissa.

a significant impact on the number of interconnections and, hence, the number of weights that must be determined, the goal generally is to obtain the smallest net consistent with the complexity of the data. Although a number of practical schemes exist for dealing with the important problem of optimal network architectures,[22-24] rigorously convergent procedures do not currently exist.

The error function is generally taken to be a sum-of-squares error function, $E = {}^1/_2\sum_i(R_i^{obs} - R_i^{net})^2$, where R_i^{obs} is the ith observed or desired value of the response function, and R_i^{net} is the corresponding value produced by the CNN. In some cases the average sum-of-squares error function, $E_{av} = (1/N)\sum_i(R_i^{obs} - R_i^{net})^2$, is sometimes used.[9] As the number of samples in the training set, N, increases, E_{av} approaches the expectation value of the sum-of-squares error, which shows its connection to the stochastic approximation employed in many statistical optimization procedures.[9]

The form of the transfer function also is important in mapping applications of CNNs. In most applications sigmoid or *tanh* transfer functions (see Figure 1) are used. Lapedes and Farber[25] have presented a very clear discussion of the function mapping characteristics of such differentiable "step functions". From their discussion it is clear that while these functions may not be optimal as basis functions to represent many input–output mappings they can, given a sufficient number of hidden nodes,[13-17] provide an adequate basis for describing a considerable variety of input–output mappings. Another aspect of sigmoid or *tanh* transfer functions is their "sharpness", which is controlled by the gain parameter, α. As seen in Figure 1, as $\alpha \rightarrow \infty$ both functions approach step functions, while as $\alpha \rightarrow 0$ both functions approach straight lines. Thus, it is expected that for most applications requiring relatively "smooth" mappings α ought to lie in a range such that the transfer function is neither too steep nor too flat (vide infra).

Optimization of the weights is generally carried out by some form of back-propagation of error, or backprop (BP), procedure.[26,27] This is a gradient-based procedure and is thus beset, as are all gradient-based procedures, by the multiple minimum problem. Hence, the likelihood of becoming "trapped" within a local minimum is very real and can make obtaining a proper solution problematic. Recently, a robust modification of this algorithm, called extended delta-bar-delta (EDBD), has been developed that holds considerable promise,[28] and it is the algorithm used here.

Several stochastic learning algorithms are also worthy of note. These are best be understood if one considers the set of weights, $\{w_i\}$, as components of a weight vector, $\mathbf{w} = (w_1, w_2, ..., w_n)^T$, where T is the transpose. Determination of the weights is then a problem of searching weight space to find a $\mathbf{w}$ that minimizes E. Various forms of simplex,[29] Monte Carlo,[30] simulated annealing,[31] and genetic[32] algorithms have all been investigated. Even differential equation-based procedures have been used.[33] While these methods appear to yield well-converged solutions, they generally take longer than BP or EDBD procedures. Recent work by Chen and Hecht-Nielsen[34] has, however, shown that, due to permutational and sign symmetries of the weight vector, the weight space is highly redundant, i.e., many weight vectors yield equally good solutions. A practical importance of this to the training of CNNs is that only a relatively small "cone" in weight space need be considered, but this requires that the constraint boundaries of the cone be known. Unfortunately, the means for determining the constraint boundaries is not yet available. Nevertheless, it does indicate that weight space is not as vast as it might appear at first view, and thus, future work in this area may point the way to more powerful new algorithms that will facilitate the rapid determination of well-converged weight vectors.

Data Representation. Data representation is a crucial part of any attempt to apply CNNs, especially to chemical problems. Generally, input to CNNs is in the form of a vector or a list of individual components. Although in many cases such an input representation is adequate, it does not capture the type of "chemical information" inherent in 2D or 3D molecular structures. In general, this problem remains unsolved, although some recent work by Kvasnicka[35,36] has addressed this problem in an interesting but limited way.

Given that vector-like input is required, the problem becomes one of choosing a set of appropriate feature descriptors. On the one hand, these descriptors should be as "global" as possible, i.e., they should be appropriate to and obtainable for a broad range of compounds. On the other hand, the set of descriptors should be as small as possible to insure that the problem of small sample size, which occurs in many chemical studies,

does not become limiting. As noted above, the number of nodes, including input nodes, determines the number of interconnections and, thus, weights that must be determined during training. It is generally considered (see, e.g., ref 33) that the number of data samples should be at least three times the number of weights, although even a factor of 3 would be considered insufficient in certain circumstances.[9] Interrelationships among the weights can, however, lower the number of *independent* weights and, thus, effectively increase the samples-to-weights ratio.

Another problem which arises in data representation is that of linear and nonlinear correlations among descriptors. When this is the case, several descriptors characterize essentially the same information. Linear correlations can be pinpointed with standard statistical correlation methods. Nonlinear correlations, however, are more difficult to uncover.[37] A recent paper by Kramer[38] presents a detailed discussion of this problem and describes a novel CNN solution based upon data encoding. Other related approaches have also been described in the literature.[39,40]

Error Analysis. A critical part of the development of any CNN is an evaluation of its performance. As generalized perceptrons are based upon supervised learning, the data set is generally divided into training and test sets, which can be obtained by appropriate random-sampling procedures if the amount of data is sufficiently large.

An ideal test set is one that spans the problem space adequately to ensure that a network which performs correctly on a test set can be considered to have "solved" the ultimate problem under study.[3] In some situations, an "acceptance test set" is kept in reserve for final network validation, but this is not feasible when the amount of data available is small.

In many cases related to chemical systems, however, the presence of small data sets requires that resampling methods must be employed—the particular method chosen will depend upon the nature of the problem and the amount and type of data available. Three methods are mainly in use today, viz. cross-validation, leave-one-out (a variant of cross-validation), and bootstrapping. A recent book by Weiss and Kulikowski[41] provides an excellent, readable discussion of these methods and their application to error estimation in generalized perceptrons as well as in other learning-based procedures. A more mathematically-oriented treatment is provided by Efron.[42,43] An alternative approach to the small data set problem has been dealt with by Stubbs,[44] who used a Bayesian prediction scheme to augment the initial data set. While this approach may not be appropriate in all cases, it certainly merits further consideration as a possible strategy for dealing with the small data sets.

The particular form of the "error function" is also important. Should all errors be treated equally? Some types of errors are more important than others. In some cases false negatives may be considerably more serious than false positives as in, for example, the diagnosis of a particular medical condition, while in other cases both types of errors may be equally serious or not very serious at all. This suggests that some weighting of the errors may be desirable or required, but how to determine appropriate weights is a very difficult task. Risk and cost functions have been used in the past, especially in Bayesian predictions,[45] but this has not generally been done in CNN studies. Nevertheless, the possibility of using them should not be precluded, and their application in specific cases may be proper and necessary.

Generalization. One of the most important attributes of CNNs is their ability to *generalize*, i.e., their ability to make reliable predictions on new data with similar accuracy to that obtained with training data. Although a number of detailed theoretical treatments of this problem have been described for binary input data,[46] to our knowledge similar treatments have not as yet been carried out for continuous inputs. The problem of generalization is related to the problem of *overfitting*. Overfitting occurs when the size of a training data set is comparable to the size of the weight space that "supports" the CNN. As the number of hidden layers and their associated nodes directly influences the number of weights, the complexity of a given network is limited by the size of the data set. Under such circumstances, the training data including noise may be fit nearly exactly, but the CNN most likely will fail on new data. Thus, generalization is best when noise is smoothed out, a situation which can be approached by obtaining more new data, by smoothing the data through averaging, and by limiting the size of the network. Examples of the latter two of these approaches are given below.

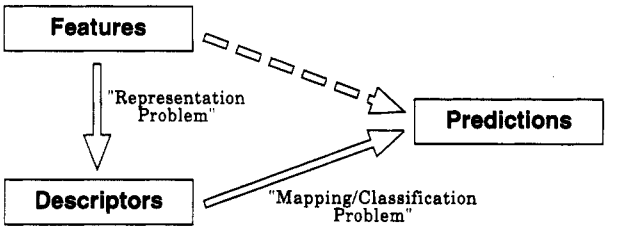


Figure 2. Schematic depiction of the prediction process (indicated by the dashed arrow) that in chemistry involves determining the value of a property (e.g., a melting point, solubility, or biological activity) from a set of relevant "chemical features". In actuality, the process is a two-step one that involves representing the chemical features by a suitable set of descriptors ("representation problem"), which are then used to predict the property ("mapping/classification problem").

RESPONSE SURFACE MODELING

The present work attempts to address a number of the above issues within the context of a model problem that, nevertheless, has characteristics of real problems of interest in the areas of quantitative structure–activity relations (QSAR) and quantitative structure–property relations (QSPR),[33,47-49] and perhaps in other areas as well. Figure 2 schematically depicts the relationship between the geometric and electronic structural features of molecules and their associated chemical properties or biological activities (see dashed arrow). Computational neural nets and methodologies typically employed in QSAR and QSPR studies follow the path designated by the two undashed arrows. The first step in the path involves generating a set of descriptors that characterize the relevant molecular features of the molecules being studied. This is the *representation problem*, which must be addressed in essentially all scientific work—CNNs generally use vector-based input. However, as noted earlier, graph-theoretical representations, which are quite natural for chemical systems, are difficult to implement in a general fashion on CNNs (cf. refs 35 and 36). We will not address this question further in the present work. The second step involves determining the mapping from the representation space to the prediction space. This is the *mapping/classification problem*. It should be noted that determining a correct input–output mapping for a complex function is a more demanding challenge than classification—the work described here is directed toward the former problem. The input–output mapping can also be interpreted as a response surface in which values of the system input variables give rise to particular system responses, and this interpretation will be used here.

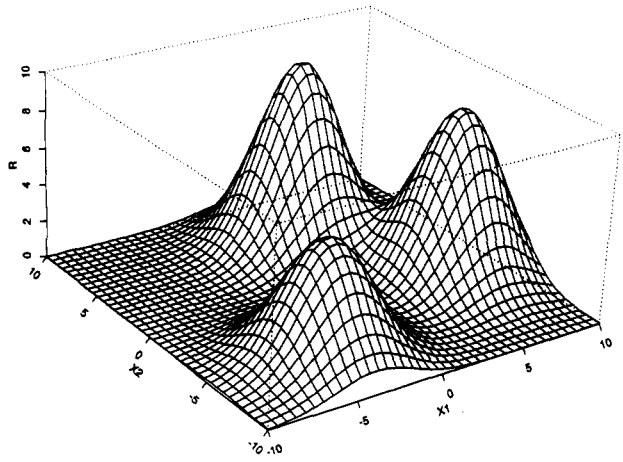


Figure 3. Three-dimensional response surface as a model for studying structure–activity and structure–property relationships. The model was constructed from three radially-symmetric Gaussian functions located at (2.0,5.0), (7.0,–2.0), and (–4.0,6.0), all with widths of 0.1 and heights of 10.0, 9.0, and 6.25, respectively. See text for further discussion.

Typically, QSAR and QSPR studies are confined to small sets of relatively similar molecules. In the future, however, it will be advantageous to take a more *global* view of the relationship of the geometric *and* electronic structural features of molecules to their biological activities and to their molecular properties such as solubility, melting point, and partition coefficients. As noted above, the relationship of biological activity to structure, for example, can be described as a *response surface*, such as the one depicted in Figure 3. In this case, the simple 3D surface is characterized by the two independent variables, x_1 and x_2, which label the coordinate axes representing molecular feature descriptors, and $R(x_1,x_2)$ represents the response surface or function, which is given as the sum of three radially-symmetric Gaussian functions located at (2.0,5.0), (7.0,–2.0), and (–4.0,–6.0), all with widths of 0.1 and heights of 10.0, 9.0, and 6.25, respectively. Each response surface corresponds to a particular type of biological activity measurement, or test system, while the region of each peak on the surface corresponds to a class of relatively similar, biologically active compounds—thus, $R(x_1,x_2)$ represents three classes of biologically active compounds.[50]

A potential benefit of the model-free approach embodied in CNNs is that there is no need to specify the functional form of the "structure–activity mapping" explicitly. Within the context of the response-surface model, a number of characteristics of the mapping ability of CNNs can be investigated. Moreover, by considering a response surface that is visualizable in three dimensions it is possible to understand the role played by various factors such as sample size, noisy and missing data, and linear and nonlinear correlations among the input variables.

A neural network with two inputs, x_1 and x_2; two hidden layers of five PEs each; and a single output PE—each PE uses the same *tanh* transfer function (see Figure 1b), which upon proper rescaling yields the output or response function $R(x_1,x_2)$.[51] Such a CNN may be designated as a 2-5-5-1 net to indicate the number of nodes in the input, hidden, and output layers, respectively. The number of weights can be determined directly from this information ($2 \times 5 + 5 \times 5 + 5 \times 1 = 40$ connection weights plus $5 + 5 + 1 = 11$ bias weights, for a total of 51 weights). The network was trained using the extended-delta-bar-delta (EDBD) modification[28] to the standard, gradient-based back-propagation of error (BP) algorithm.[26,27] This procedure was found to produce a CNN that represents the response surface quite accurately (RMS error = 0.29) when trained on a uniform 30×30 grid of 900 sample points. Although it is possible to realize essentially all nonpathological mappings with a single layer of hidden units[13-17] (vide supra), Lapedes and Farber[25] have shown that two hidden layers are more efficient than one. Their work suggests that a "bump" function, which closely resembles a multidimensional Gaussian, can be modeled with five PEs arranged in two hidden layers, the first layer containing four PEs and the second layer containing a single PE. This relationship does not, however, necessarily extend to the "three-Gaussian" response surface studied here since the sigmoid or *tanh* transfer function of a given PE may contribute to the description of more than one Gaussian "bump".

A uniform 40×40 grid of 1600 test points was used to evaluate the accuracy of the mapping produced by the 2-5-5-1 net, and the results showed that the net was capable of reproducing the true response surface to within an RMS error of 0.38. Several other network topologies containing more PEs in the hidden layers were also investigated (vide infra), but the 2-5-5-1 net was used in essentially all subsequent experiments due to its relative simplicity and its ability to represent the response surface accurately (i.e., generalize) even for relatively small sets of training data. Most importantly, it should be recognized that the "testing grid" of 1600 points represents a very stringent test of the 2-5-5-1 net's ability to accurately predict the response surface over a much larger domain of the input variables than is usually the case in real QSAR or QSPR studies, where a rather limited set of test samples is typically considered.

Sample Size. For many interesting QSAR and QSPR problems only a limited set of compounds is generally available for training and testing a CNN, typically 10–50, although sometimes even as many as several hundred may be available. Since statistical considerations suggest a minimum ratio of three samples per weight[33] it is often difficult in practice to separate the effects of inadequate training data sets from possible limitations in the mapping ability of a given CNN. The response-surface model studied here, where the actual function to be modeled is known, allows one to choose data sets of arbitrary size so that issues of sample size versus number of network weights can be studied in great detail.

The original training set of 900 evenly distributed points had a samples-to-weights ratio of approximately 17:1. Three smaller training sets containing 156, 104, and 52 sample points (see Figure 4), with samples-to-weights ratios of 3:1, 2:1, and 1:1, respectively, were chosen to investigate the effect of sample size on the ability of the 2-5-5-1 net to accurately represent the response surface. These training sets were chosen such that they approximate distributions of compounds that might occur in real QSAR studies. When an active lead compound is found, the "chemical space" near the lead is explored thoroughly, while when an inactive analog is produced, that area of the chemical space is avoided if possible. Accordingly, for each training set, one-fourth of the points were taken from a uniform random distribution in the square region surrounding each of the three peaks of active compounds, and the remaining one-fourth of the sample points were taken from a uniform random distribution over the entire domain of the response function shown in Figures 3 and 4.

The results given in Table I show that even when the samples-to-weights ratio was 1:1, the 2-5-5-1 network did a reasonable job of "learning" the response function. Visual inspection of the response surfaces depicted in Figure 5, which were obtained from sample sets of 900, 156, 104, and 52 data points,

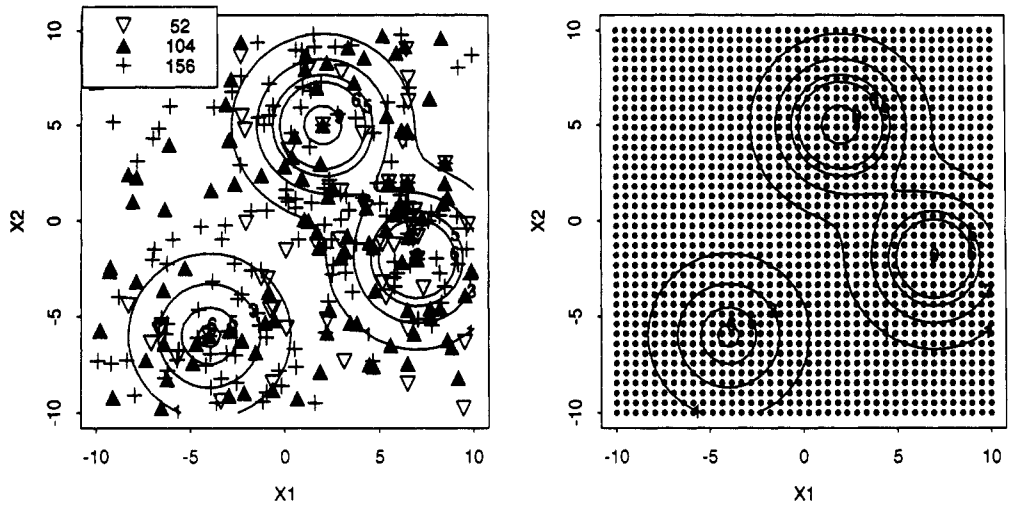


Figure 4. (a) Distribution of the sample points used in the present study. Three sets containing 156, 104, and 52 sample points, respectively, were chosen from random uniform distributions as described in the text. The key in the figure shows the graphical designation given to the points in each of the three sets. (b) Uniform distribution of a 40 × 40 grid of 1600 points used as a testing set.

Table I. Characteristics of Feedforward Neural Net Descriptions of Three-Gaussian Response-Surface Model

no. of data points training set	characteristics of data points	network architecture	samples/ weights ratio	RMS error training set	RMS error test set
900		2-5-5-1	17:1	0.29	0.38
156		2-5-5-1	3:1	0.35	0.49
104		2-5-5-1	2:1	0.48	0.78
52		2-5-5-1	1:1	0.35	0.94
156	noisy[a]	2-5-5-1	3:1	0.77	0.79
52	noisy[a]	2-5-5-1	1:1	0.80	1.72
52	noisy[a,b] (averaged)	2-5-5-1	1:1	0.52	1.17
156	noisy[a]	2-40-40-1	1:12	0.57	1.02
156	random[c]	3-5-5-1	~3:1	0.28	0.47
156	correl[d]	3-5-5-1	~3:1	0.27	0.40

[a] Gaussian noise[53] was added to the response function (see Figure 7a)—see text for details. [b] Five noisy replicates at each data point were averaged, and the averaged values were then used to train the CNN. See text for details. [c] A third independent variable uncorrelated with x_1 and x_2 was added. See text for further discussion. [d] A third independent variable quadratically correlated with x_2 was added. See text for further discussion.

respectively, clearly shows that all four surfaces are *qualitatively* of the same shape as the "test" surface depicted in Figure 3. All of the major features of this surface are accounted for by the surfaces in Figure 5, even the surface obtained from the 52-point training set, which had approximately the same number of samples as weights.

Figure 6a shows the output from each of the five PEs in the first hidden layer of the 2-5-5-1 net trained on 156 data points. Each of the surfaces, which resemble a three-dimensional "shelf," are generated from linear combinations of the two inputs (suitably weighted) that are then "passed through" the nonlinear, *tanh* transfer function (see Figure 1b) in each PE (cf. ref 25). Figure 6b shows the output from each of the five PEs in the second hidden layer. These more complicated surfaces are obtained, as were those in the first hidden layer, from linear combinations of surfaces generated in the first hidden layer that are then passed through the nonlinear *tanh* transfer function in each of the PEs in the second hidden layer. The final surface, i.e., the response surface of interest, is generated by the single PE in the output layer, which again appropriately combines the outputs from the PEs in the second hidden layer and outputs this value through the nonlinear function of the output PE. The output is then rescaled to provide the desired response surface.

Transfer Functions. As noted earlier, the form of the transfer functions can be altered by the choice of the gain parameter, α (see Figure 1b). Generally, the gain parameter is taken to be $\alpha = 1.0$, and this is the value used in essentially all the studies carried out in this work. However, two 2-5-5-1 CNNs were investigated in which the gain parameter was taken to be $\alpha = 10.0$ ("step function") and $\alpha = 0.1$ (approximately a "straight line"), respectively. In both cases, learning did not converge, which is not surprising given the shape of the three-Gaussian response surface. In some cases radial Gaussians have been used as transfer functions.[52] Based on the earlier discussion of Lapedes and Farber regarding "bump" functions,[25] it is expected that radial Gaussians can provide a suitable "basis" for representing many input–output mappings of interest, especially surfaces similar to the response surface studied here. Nevertheless, we have chosen to carry out our work with the more "traditional" *tanh* transfer function.

Noisy Data. The experiments reported above were conducted in the absence of noise, but in real situations noise is present to some degree in all measurements. Accordingly, the noisy response surface depicted in Figure 7a was generated by perturbing the smooth response surface with Gaussian noise.[53] Examination of the figure shows that while the general topography of the response surface remains, some of its more subtle features are obliterated.

Figure 7b shows the response surface of a 2-5-5-1 net trained on 156 noisy points. It is clear from the figure that the general features of the response surface are reproduced here. As the number of sample points is decreased, the response surface becomes distorted, as is seen in FIgure 7c for a 2-5-5-1 net trained on 52 noisy data points. In real experimental situations it is not always feasible or desirable to increase the number of data points, especially if each point represents a different compound. In such cases, standard statistical averaging procedures can help remove some of the "experimental noise", a procedure which is quite feasible and highly desirable in most experimental situations. This is illustrated in Figure 7d, where the shape of the response surface of a 2-5-5-1 net trained on 52 *averaged* data points is considerably improved

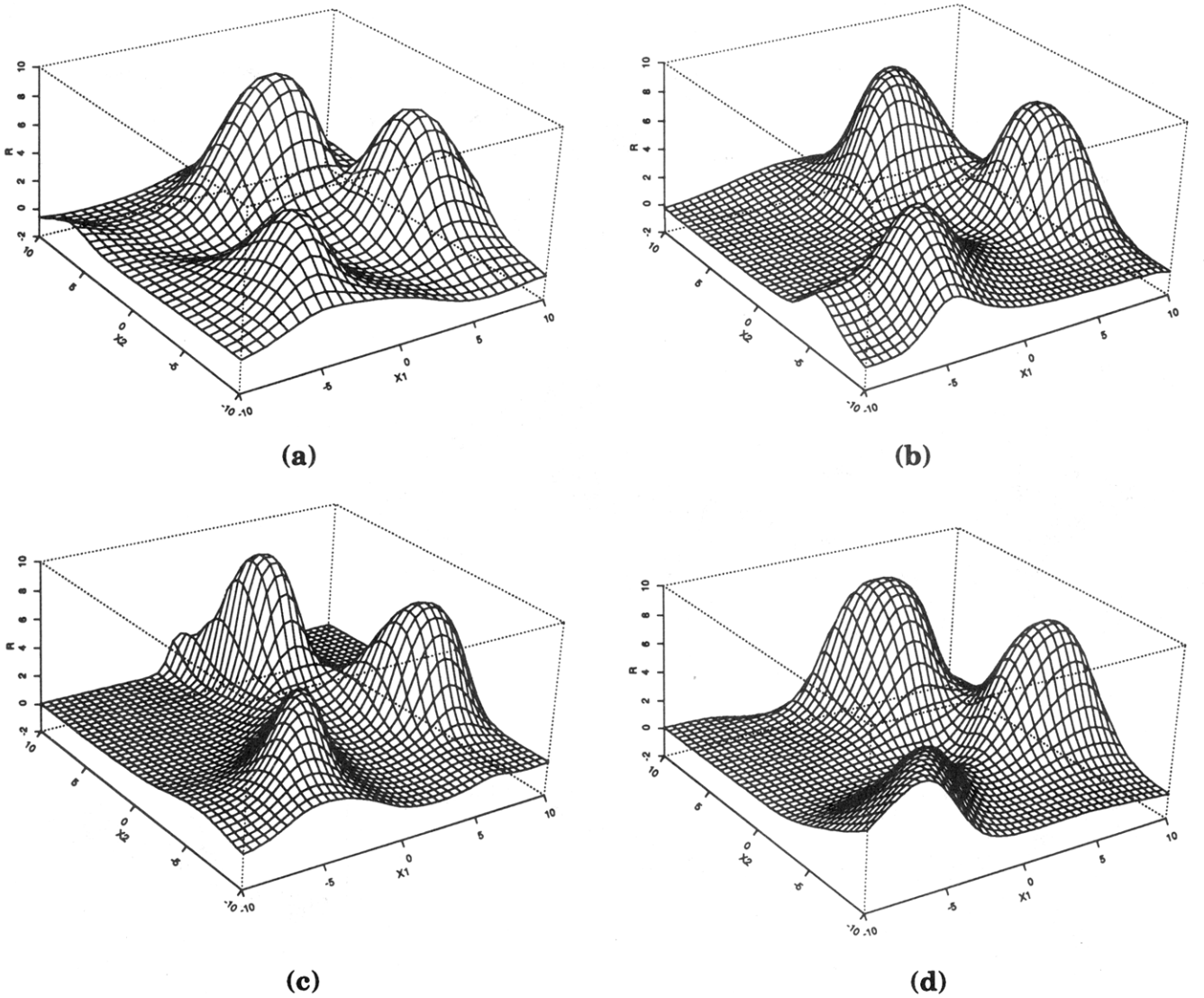


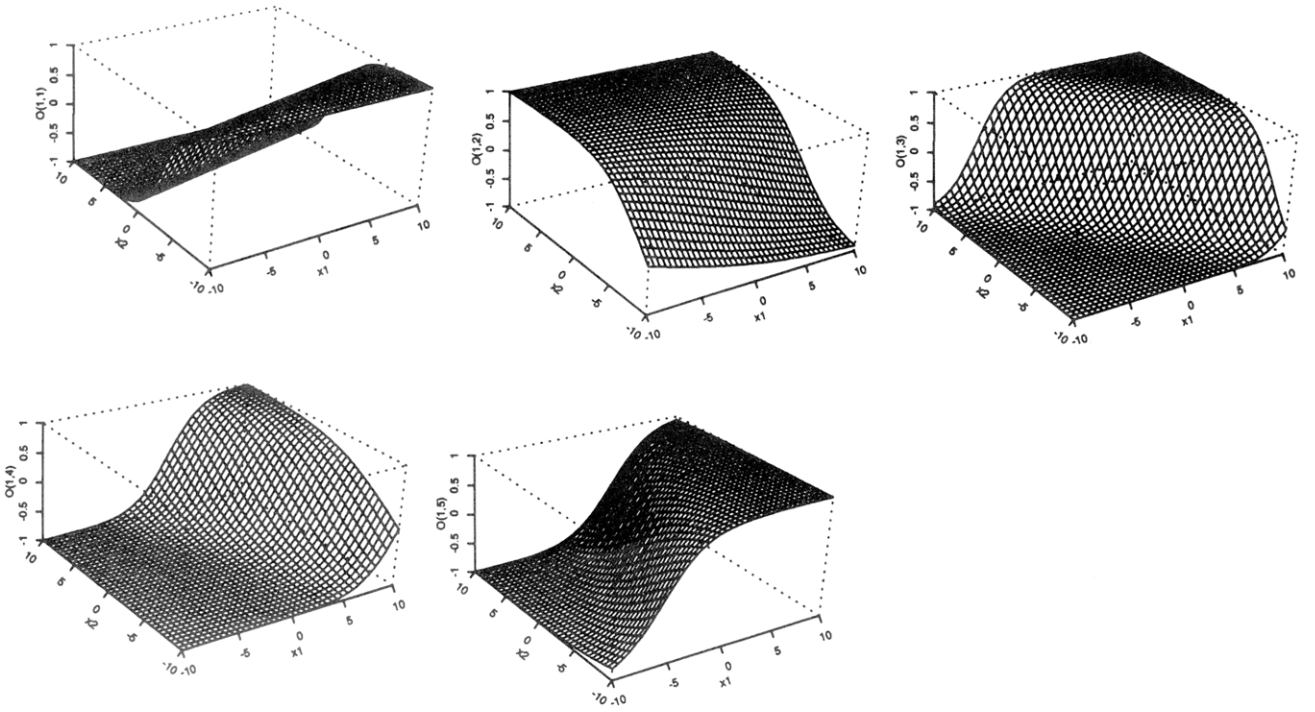
Figure 5. (a) Response surface obtained from a 2-5-5-1 net trained on a uniform 30 × 30 grid of 900 noise-free sample points. (b) Response surface obtained from a 2-5-5-1 net trained on 156 noise-free sample points. (c) Response surface obtained from a 2-5-5-1 net trained on 104 noise-free sample points. (d) Response surface obtained from a 2-5-5-1 net trained on 52 noise-free sample points. All four response functions were tested on a uniform 40 × 40 grid of 1600 sample points.

over the surface shown in Figure 7c and approaches the shape of the surface obtained from an identical net trained on 52 non-noisy data points (see Figure 5d). Each of the averaged data points was obtained by averaging five replicates with Gaussian noise at each of the 52 points.

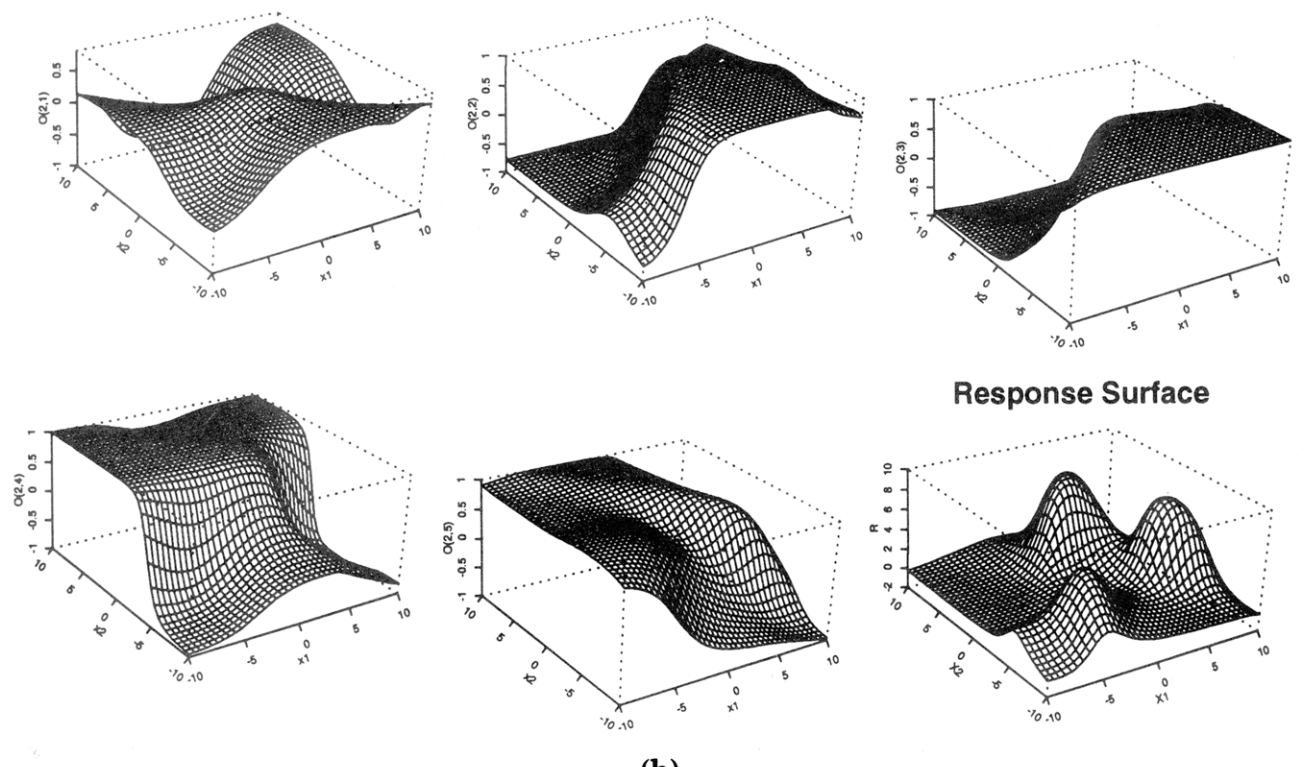
As shown in Table I, a 2-40-40-1 net with 1801 weights (approximately 1 sample per 12 weights) trained on 156 noisy data points yielded an RMS error of 0.57, which was less than the value of 0.77 for the corresponding 2-5-5-1 net. On the test set, the 2-40-40-1 net had an RMS error of 1.02, which was poorer than the corresponding value for the 2-5-5-1 net. This behavior is expected and indicates that the while the larger net, due to its greater complexity, was able to "learn" the noisy data better it was less able to generalize correctly. Figure 7e depicts the response surface obtained from the 2-40-40-1 net. The highly "ruffled" surface is clearly indicative of the net's ability to learn the noise in the data.

The fact that the simpler 2-5-5-1 net was able to model the response surface with 156 noisy data points and generalize from the test set (see Table I) may be due to the *relative* simplicity and smoothness of the surface investigated here (see Figure 3). A more complex and highly-variable function would provide a more stringent test of a CNN's ability to generalize. More work is needed in this area to clarify the relationship of network complexity to sample size.

Representation Issues. As noted earlier, finding a suitable representation to treat a given problem or class of problems is a demanding and difficult task. Finding the *optimum* representation is an even more daunting task. Chemical systems, in particular, provide a significant challenge due to the fact that their "natural" representation is in the form of chemical graphs.[54] Vector-based representations, as illustrated by the example given here, are employed in most CNNs and in essentially all cases investigated to date based upon generalized perceptrons. In the example considered in this work, the precise nature of the two independent, descriptor variables, x_1 and x_2, was not considered. In real applications, however, this is generally one of the major problems confronting the researcher trying to investigate QSAR, QSPR, or related problems, especially in cases where it is desired that the independent, descriptor variables provide a characterization of a significant portion of the "chemical universe". Typical issues that must be addressed by researchers regarding the choice of descriptor variables include the nature of the variables, whether they can be calculated or must be determined experimentally (and thus whether values are avail-



(a)



(b)

Figure 6. (a) Outputs from the five PEs of the first hidden layer of a 2-5-5-1 net trained on 156 noise-free sample points. The notation $O(i,j)$ on the ordinate denotes the ith hidden layer and the jth PE in that layer, respectively: $i = 1$ and $j = 1, 2, ..., 5$ in this case. (b) Outputs from the five PEs of the second hidden layer of the same 2-5-5-1 net [i.e., $O(2,j)$, $j = 1, 2, ..., 5$] and from the response function generated by the net (bottom right surface—note the change of scale on the vertical axis).

able—the "missing data problem"), the number of variables required to fully characterize the system under study, and whether dependencies among the variables exist due to linear or nonlinear correlations.

We have investigated several of these issues in terms of the response-surface model system described above—the results are summarized in Table I. For example, when a third input variable, x_3, was added to the system but taken to be random (i.e., uncorrelated with x_1, x_2, and R), all the weights of the trained CNN connecting x_3 to the nodes in the first hidden layer were zero, and the RMS error of the net was comparable to that trained on only two input variables. When x_3 was taken to be nonlinearly correlated with x_2 (i.e., $x_3 = {}^1/_2x_2^2$), the 3-5-5-1 net learned and generalized as well as the net trained on just x_1 and x_2. These results suggest that a CNN's ability to model a response surface is not materially altered

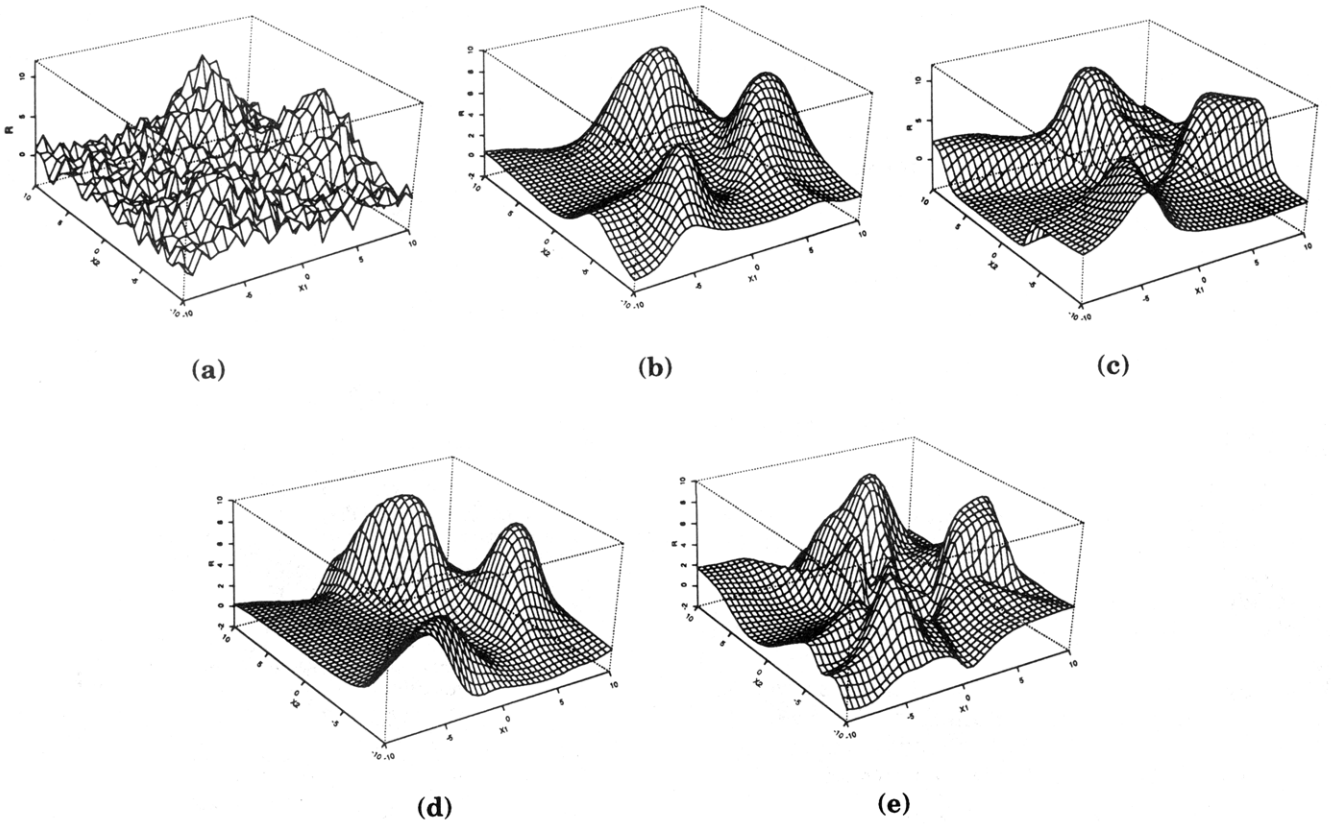


Figure 7. (a) Model response surface with Gaussian noise (see text for details). (b) Response surface obtained from a 2-5-5-1 net trained on 156 noisy sample points. (c) Response surface obtained from a 2-5-5-1 net trained on 52 noisy sample points. (d) Response surface obtained from a 2-5-5-1 net trained on 52 averaged sample points (see text for details). (e) Response surface obtained from a 2-40-40-1 net trained on 52 noisy sample points.

due to extraneous input variables nor by correlations among input variables, as long as the input variables contain sufficient information that is relevant to the mapping being represented. Nevertheless, it is desirable to reduce the dimensionality of the input space as much as possible so that the complexity of the CNN does not needlessly become too great. Methods for reducing the dimensionality of the input space were noted earlier[38-40] and will not be discussed further here.

OVERVIEW AND CLOSING REMARKS

Generalized perceptrons can function as powerful, essentially model-free, mapping devices that can be applied to a wide range of problems in chemistry.[55] The power of these CNNs lies in their ability to represent very general mappings without the need to specify the mathematical form of the mapping explicitly. And although the model three-dimensional response surface with two input variables given here is relatively simple, it nonetheless illustrates and clarifies a number of important issues that are relevant to the applications of generalized perceptrons in general and in chemistry in particular. These issues include choice of an appropriate data representation, selection of a suitable learning algorithm for determining the network weights, and evaluation of the network's ability to fit training and test data (i.e., to generalize) effectively. In addition, problems due to small sample size (relative to the number of weights), which plague many chemical applications of CNNs, must be dealt with. However, the above example shows that even in cases where the number of samples relative to the number of weights is less than the minimum number required, i.e., three,[33] generalized perceptrons can still be trained to provide a reasonably accurate representation of the desired response surface (see Table I). Earlier applications of generalized perceptrons to chemical reaction prediction[56,57] showed that reasonable generalization could be obtained from nets trained on a minimal number of samples, even when the number of samples was less than the number of weights. In such cases two important points should be made. First, in many chemical studies only a relatively small number of compounds may potentially exist (i.e., certain molecular structures are inherently unstable and cannot be studied using normal chemical methods), and hence, predictions made in such cases can only be based on limited data. Second, knowledge of structural as well as other types of chemical information can be of assistance in choosing a "representative" training set. While such an approach admittedly introduces a bias into the learning process, it nonetheless may provide a satisfactory means for studying chemical systems where only sparse data exists and classical statistical methods are not applicable. Statistical methods such as "leave-one-out" or bootstrapping[42-44] can, in some cases, provide the means for addressing the "small sample problem", but the power of these methods has not been fully exploited in CNN applications to date. Additional work is needed in this area.

Once a CNN is sufficiently trained for a particular QSAR or QSPR application, in some cases it may be of interest to determine the location of extrema of interest on the response surface—whether the extrema are maxima or minima depends upon the particular application. In such cases these extremal points represent potentially active compounds (maxima) or compounds with maximal or minimal values for a given property. This more global approach to QSAR or QSPR represents an important new direction for CNNs and will provide a potent new tool for use in computer-aided molecular design.

In low-dimensional cases, such as the one illustrated here for two-dimensions, it is possible to locate extrema by evaluating the response function on a uniform grid. For higher-dimensional cases this is not feasible, and other optimization methods must be used. As the response surfaces generally possess numerous extrema, gradient-based methods are doomed to failure. However, methods such as those based upon simulated annealing[31] or genetic algorithms,[32] which were discussed earlier with regard to the training of CNNs, are also applicable here, and only a slight modification of these algorithms is needed to adapt them to the current problem.

Many new applications of CNNs remain to be discovered and exploited in chemistry and related fields. The rapidly growing literature in these areas—it has increased more than 10-fold in the last 3 years—attests to their vitality. Moreover, the breadth and variety of the applications are increasing as well. Whether CNNs will fulfill their promise in chemistry at this time still, however, remains an open question. Clearly, CNNs can be used to build complicated models directly from data. What is not clear is whether the models they generate are significantly better than could be constructed by other means. However, even if the answer to the above question is no in some cases, the power of CNNs to address a wide and growing variety of problems in many fields strongly suggests that CNNs warrant further serious study as problem-solving tools and as interesting entities in their own right.

REFERENCES AND NOTES

(1) Rumelhart, D. E.; McClelland, J. L. *Parallel Distributed Processing*; MIT Press: Cambridge, MA, 1986; Vols. I and II.

(2) Hertz, J.; Krogh, A.; Palmer, R. G. *Introduction To The Theory of Neural Computation*; Addison-Wesley Publishing Co.: Redwood City, CA, 1991.

(3) Hecht-Nielsen, R. *Neurocomputing*; Addison-Wesley Publishing Co.: Redwood City, CA, 1990.

(4) Simpson, P. K. *Artificial Neural Systems: Foundations, Paradigms, Applications, and Implementations*; Pergamon Press: New York, 1990.

(5) Lippmann, R. P. An Introduction to Computing with Neural Nets. *IEEE Trans. Acoust., Speech, Signal Process.* **1987**, *35*, 4–22.

(6) Although neural networks are inherently, highly-parallel computing devices, they have up to now mostly been implemented in software on normal scalar computers and not in silicon chips.

(7) In some CNNs, such as Hopfield nets, the values of the weights are predetermined, and, thus, these nets do not "learn" in the sense described here.

(8) Perceptrons have only two layers of nodes—an input and an output layer—and can only handle linearly separable problems; see refs 1–5.

(9) White, H. Learning in Artificial Neural Networks: A Statistical Perspective. *Neural Comp.* **1989**, *1*, 425–464.

(10) See Chapter 5 of Wasserman, P. D. *Neural Computing–Theory and Practice*; Van Nostrand Reinhold: New York, 1989, for a very clear description of stochastic learning procedures.

(11) The transfer function is also called an *activation function* or "*squashing function*," the latter because of the form of the sigmoid or *tanh* functions (see Figure 1) used in most CNN applications.

(12) The requirement that f be continuous is necessitated by the use of gradient-based learning algorithms such as back-propagation, which use derivatives. Other learning algorithms based on nonderivative procedures such as simulated annealing or genetic algorithms do not require continuous transfer functions.

(13) Kolmogorov, A. N. On the Representation of Continuous Functions of Several Variables by Superposition of Continuous Functions of One Variable and Addition. *Dokl. Akad. Nauk SSSR* **1957**, *114*, 953–956.

(14) Cybenko, G. Approximation by Superpositions of a Sigmoidal Function. *Math. Contr. Sig. Sys.* **1989**, *4*, 303–314.

(15) Hecht-Nielsen, R. Theory of the Backpropagation Neural Network. In *Proceedings of the International Joint Conference on Neural Networks*; IEEE Press: New York, 1989; Vol. I.

(16) Stinchcombe, M.; White, H. Universal Approximation Using Feedforward Networks with Non-Sigmoid Hidden Layer Activation Functions. In *Proceedings of the International Joint Conference on Neural Networks*, Washington, DC, June 1989; IEEE Press: New York, 1989; Vol. I, pp 607–611.

(17) Chester, D. L. Why Two Hidden Layers are Better Than One. In *Proceedings of the International Joint Conference on Neural Networks*, Washington, DC, Jan 1989; IEEE Press: New York, 1989; Vol. I, pp 265–268.

(18) Kosko, B. *Neural Networks and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence*; Prentice Hall: Englewood Cliffs, NJ, 1992.

(19) While this may be true technically, it omits the fact that the network topology, i.e., the number of hidden layers and the number of nodes in each layer, as well as the type of transfer function must at some point be specified, and this may be construed, in some sense, as being analogous to choosing a model.

(20) Favlow, S. J., Ed. *Self-Organizing Methods in Modeling: GMDH Algorithms*; Marcel Dekker: New York, 1984.

(21) See Chapter 5 of ref 2 for a detailed discussion of alternative error functions.

(22) Mézard, M.; Nadal, J.-P. Learning in Feedforward Layered Networks: The Tiling Algorithm. *J. Phys. A: Math. Gen.* **1989**, *22*, 2191–2204.

(23) Marchand, M.; Golea, M.; Ruján, P. A Convergence Theorem for Sequential Learning in Two-Layer Perceptrons. *Europhys. Lett.* **1990**, *11*, 487–492.

(24) Frean, F. The Upstart Algorithm: A Method for Constructing and Training Feedforward Neural Networks. *Neural Comp.* **1990**, *2*, 198–209.

(25) Lapedes, A.; Farber, R. How Neural Nets Work. In *Neural Information Processing Systems*; Andersen, D. Z., Ed.; American Institute of Physics: New York, 1988; pp 442–456.

(26) Werbos, P. J. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences. Doctoral* Dissertation. Harvard University, Cambridge, MA, Nov 1974.

(27) Rumelhart, D. E.; Hinton, G. E.; Williams, R. J. Learning Representations by Backpropagating Errors. *Nature* **1986**, *323*, 533–536.

(28) Minai, A. A.; Williams, R. D. Acceleration of Back-Propagation Through Learning Rate and Momentum Adaptation. In *Proceedings of the International Joint Conference on Neural Networks*, Jan 1990; Caudill, M., Ed.; Lawrence Erlbaum Assoc.: Hillsdale, NJ, 1990; pp 676–679.

(29) Fletcher, R. *Practical Methods of Optimization*; Wiley-Interscience: New York, 1980; Vol. 1.

(30) Baba, N. A New Approach to Finding the Global Minimum of the Error Function of Neural Networks. *Neural Networks* **1989**, *2*, 367–373.

(31) Szu, H.; Hartley, R. Fast Simulated Annealing. *Phys. Lett.* **1987**, *122*, 157–162.

(32) Montana, D. J.; Davis, L. Training Feedforward Networks Using Genetic Algorithms. In *Eleventh International Joint Conference on Artificial Intelligence*, Detroit, 1989; Sridharan, N. S., Ed.; Morgan-Kaufmann: San Mateo, CA, 1989; pp 762–767.

(33) 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.

(34) Chen, A. M.; Hecht-Nielsen, R. On the Geometry of Feedforward Neural Network Weight Spaces. In *Second International Conference on Artificial Neural Networks*; IEE: London, 1991; pp 1–4.

(35) Kvasnicka, V. An Application of Neural Networks in Chemistry. Prediction of ^{13}C NMR Chemical Shifts. *J. Math. Chem.* **1991**, *6*, 63–76.

(36) Kvasnicka, V.; Pospichal, J. Application of Neural Networks in Chemistry. Prediction of Product Distribution of Nitration in a Series of Monosubstituted Benzenes. *J. Mol. Struct. (THEOCHEM)* **1991**, *235*, 227–242.

(37) Sharaf, M. A.; Illman, D. L.; Kowalski, B. R. *Chemometrics*; Wiley-Interscience: New York, 1986.

(38) Kramer, M. A. Nonlinear Principal Component Analysis Using Autoassociative Neural Networks. *AIChE J.* **1991**, *37*, 233–243.

(39) See Chapter 6 of ref 2 for a discussion of data encoding and dimensionality reduction schemes.

(40) Hussain, A.; Kane, A. A. Neural Network Approach for Identification of Relevant Structural Descriptors in QSAR Analysis. Presented at the Midwest Meeting of the American Association of Pharmaceutical Scientists, Chicago, IL, May 9, 1992.

(41) Weiss, S. M.; Kulilowski, C. A. *Computer Systems That Learn*; Morgan-Kaufman: San Mateo, CA, 1991.

(42) Efron, B.; Gong, G. A Leisurely Look at the Bootstrap, the Jackknife, and Cross-Validation. *Am. Stat.* **1983**, *37*, 36–48.

(43) Efron, B. *The Jackknife, the Bootstrap and Other Resampling Plans*; SIAM: Philadelphia, 1982.

(44) Stubbs, D. F. Multiple Neural Network Approaches to Clinical Expert Systems. *SPIE* **1990**, *1294*, 433–441.

(45) Duda, R. O.; Hart, P. E. *Pattern Classification and Scene Analysis*; Wiley-Interscience: New York, 1973.

(46) See Chapter 6 of ref 2 for a detailed discussion of many aspects of generalization in CNNs.

(47) Aoyama, T.; Suzuki, Y.; Ichikawa, H. Neural Networks Applied to Quantitative Structure/Activity Relationships. *J. Med. Chem.* **1990**, *33*, 2583–2590.

(48) Rose, V. S.; Croall, I. F.; MacFie, H. J. H. An Application of Unsupervised Neural Network Methodology (Kohonen Topology-Preserving Mapping) to QSAR Analysis. *Quant. Struct.–Act. Relat.* **1991**, *10*, 6–15.

(49) Hussain, A. S.; Yu, X.; Johnson, R. D. Application of Neural Computing in Pharmaceutical Product Development. *Pharm. Res.* **1991**, *8*, 1248–1252.

(50) The situation modeled here by the simple "three-Gaussian" response surface is quite prevalent in actual studies of biologically active

compounds, where different, seemingly unrelated, classes of molecules may possess similar biological activity in a given test system.
(51) As the *tanh* transfer function used here "squashes" the output such that $-1 \leq R \leq 1$, R must be rescaled to produce an appropriate value for the function.
(52) Moody, J.; Darken, C. Learning with Localized Receptive Fields. In *Proceeding of the 1988 Connectionist Summer School*, Pittsburgh, 1988; Touretzky, D. S., Hinton, G. E., Sejnowski, T. J., Eds.; Morgan-Kaufmann: San Mateo, CA, 1988; pp 133–143.
(53) The noisy data sets were generated by adding Gaussian noise, with a mean of 0 and a standard deviation of 1, to the value of the response variable for the 900-, 156-, and 52-point data sets.
(54) Trinajstić, N. *Chemical Graph Theory*; CRC Press: Boca Raton, FL, 1983; Vols. I and II.
(55) Zupan, J.; Gasteiger, J. Neural Networks: A New Method for Solving Chemical Problems or Just a Passing Phase? *Anal. Chim. Acta* **1991**, *248*, 1–30.
(56) Elrod, D. W.; Maggiora, G. M.; Trenary, R. G. Applications of Neural Networks in Chemistry. 1. Prediction of Electrophilic Aromatic Substitution Reactions. *J. Chem. Inf. Comput. Sci.* **1990**, *30*, 477–484.
(57) Elrod, D. W.; Maggiora, G. M.; Trenary, R. G. Applications of Neural Networks in Chemistry. 2. A General Connectivity Representation for the Prediction of Regiochemistry. *Tetrahedron Comput. Methodol.* **1990**, *4*, 163–174.