Higher-order probabilistic perceptrons as Bayesian inference engines

J. W. Clark,[1] K. A. Gernoth,[2,3] S. Dittmar,[4] and M. L. Ristig[4]
[1]*McDonnell Center for the Space Sciences and Department of Physics, Washington University, St. Louis, Missouri 63130*
[2]*Department of Physics, UMIST, Manchester M60 1QD, United Kingdom**
[3]*International School for Advanced Studies (SISSA/ISAS), I-34014 Trieste, Italy*
[4]*Institut für Theoretische Physik, Universität zu Köln, D-50937 Köln, Germany*
(Received 8 September 1998)

An explicit structural connection is established between the Bayes optimal classifier operating on K binary input variables and a corresponding two-layer perceptron having normalized output activities and couplings from input to output units of all orders up to K. With suitable modification of connection weights and biases, such a higher-order probabilistic perceptron should in principle be able to learn the statistics of the classification problem and match the *a posteriori* probabilities given by Bayes optimal inference. Specific training algorithms are developed that allow this goal to be approximated in a controlled variational sense. An application to the task of discriminating between stable and unstable nuclides in nuclear physics yields network models with predictive performance comparable to the best that has been achieved with conventional multilayer perceptrons containing only pairwise connections. [S1063-651X(99)10205-8]

PACS number(s): 07.05.Mh, 02.50.Ph, 07.05.Tp, 07.05.Kf

I. INTRODUCTION

The purpose of this paper is to establish and exploit a structural relationship between feedforward neural networks of a certain type and Bayes' rule of inference. This connection was already present in seminal form in the original book of Minsky and Papert [1] and is quite near the surface in the classic work of Duda and Hart [2]. A similar investigation was made more recently by Stolorz *et al.* [3], based on a Bahadur decomposition [4] of the class-conditional probability. Among other contemporary works exploring interesting relations and comparisons between neural networks and Bayesian statistics [5–16], those of Ruck *et al.* [9], Wan [10], and Richard and Lippmann [11] are most relevant to our considerations. These authors show that conventional neural-network techniques yield architecture-limited approximations to the *a posteriori* probabilities of Bayes optimal classifiers. Here we shall examine the structure of the output generated by a two-layer perceptron that involves (i) normalized, soft-maximum (soft-max) activation or ''squashing'' functions and (ii) arbitrary higher-order couplings to each output unit from the inputs, along with the standard complement of biases and pairwise connections. Such systems will be called higher-order probabilistic perceptrons (HOPPs). For binary inputs and finite input and output spaces, it will be demonstrated that the finite HOPP architecture is sufficiently general to embody the full statistical correlations inherent in the Bayesian approach [2] to classification problems. Appealing to the formal results from Refs. [9–11] on the training of feedforward networks to approximate Bayesian inference, we develop supervised learning rules which, given an adequate body of training examples, will enable HOPP networks to produce close estimates of Bayesian *a posteriori* probabilities.

Neural networks are of fundamental interest within physics as archetypes of dynamical and statistical systems that can learn by example, compute, and perform statistical inference. Their conceptual importance will grow as information theory and Bayesian statistics assume greater roles within mainstream physics. A more practical facet of neural networks is seen in their emergence as effective tools for data analysis in astronomy and high-energy physics and for the statistical modeling of complex systems such as proteins, genes, and nuclei [17,18]. The present investigation is concerned with both formal and applied aspects of neural-network theory.

In previous work [18–22], custom-tailored multilayer feedforward neural networks have been applied successfully to a variety of classification and function approximation problems in nuclear physics. With the proton and the neutron numbers as binary-encoded input variables, global network models have been constructed that capture the statistical regularities of the stability-instability dichotomy [19–21], ground-state spins and parities [17,21,20], atomic masses [18–21], and branching probabilities for different decay modes [22]. Networks trained with error-backpropagation schemes [23–26] based on gradient-descent minimization of appropriate objective functions can achieve predictive accuracy competitive with that of traditional phenomenological models. Continuing in a similar vein, we shall use the stability-instability classification problem as a test of the practicality and the generalization abilities of higher-order probabilistic perceptrons. In contrast to conventional, multilayer networks with exclusively pairwise connections, allowance will be made for the presence of feedforward connections of any order between input and output units.

Formal analysis of the properties of HOPP networks is conducted in Secs. II–VI. In Sec. II we introduce the standard pattern-classification problem and recall the Bayesian strategy for its solution. The higher-order probabilistic perceptron is defined in Sec. III, and its structural relationship to Bayes' rule of inference is delineated in Sec. IV. Explicit formulas linking the network weights to the priors and class-

*Permanent address.

conditional probabilities that enter Bayes' rule are given in Sec. V. Section VI collects a number of remarks intended, in part, to illuminate the importance of the structural identity established in Sec. IV. The remainder of the paper is concerned with actual implementation of HOPP networks as ''Bayesian inference engines.'' Learning algorithms for the determination of HOPP connection weights are developed in Sec. VII, based on mean-square-error and relative-entropy objective functions. The results of numerical application of HOPP networks to the nuclear stability-instability discrimination problem are presented and discussed in Sec. VIII. In Sec. IX, concluding remarks are directed to the optimization of HOPP architectures and comparisons with traditional multilayer models.

II. PROBLEM FORMULATION AND BAYESIAN DECISION THEORY

In the standard pattern-classification problem [2,11], one faces the task of assigning individual patterns $\mathbf{x}=(x_1,\cdots,x_K)$ of finite length K to one of L classes $\lambda=1,\ldots,L$. The ''input'' variables x_k may be continuous or binary. However, the analysis to be performed in Sec. IV will be restricted to the binary case, i.e., $x_k\in\{0,1\}$ for $k=1,\ldots,K$, so that the input patterns are bit strings. In a more general formulation of the classification problem, a given input pattern may be assigned to more than one category λ. Aside from peripheral comments in Sec. VI, we shall not be concerned with this elaboration.

The Bayesian approach to the standard classification problem [2] is explicitly probabilistic and rests on the construction of the *a posteriori* probability $P(\lambda|\mathbf{x})$ that the class is λ if the input pattern is known to be $\mathbf{x}$. The probability of error is minimized by choosing the class λ for which $P(\lambda|\mathbf{x})$ assumes its largest value, a decision principle which defines the *Bayes optimal classifier*. The *a posteriori* probability is constructed via Bayes' rule

$$P(\lambda|\mathbf{x})=\frac{p(\mathbf{x}|\lambda)P(\lambda)}{p(\mathbf{x})}, \tag{1}$$

in terms of the class-conditional probability (or likelihood) $p(\mathbf{x}|\lambda)$ that the pattern is $\mathbf{x}$ when the category is known to be λ and the *a priori* probability $P(\lambda)$ of finding λ. The denominator $p(\mathbf{x})=\sum_{\mu=1}^{L}p(\mathbf{x}|\mu)P(\mu)$ guarantees that $\sum_{\lambda=1}^{L}P(\lambda|\mathbf{x})=1$, as required for a probability distribution over exhaustive outcomes. [Strictly, of course, $p(\mathbf{x}|\lambda)$ is a probability density rather than a probability in the case that the input variables x_k are continuous.]

Within this framework, one may distinguish two types of classification problems, namely, *deterministic* and *probabilistic*. In a deterministic problem, knowledge of the input vector $\mathbf{x}$ is in principle sufficient to determine the class λ unambiguously, i.e., there exists a mapping $\nu(\mathbf{x}):\mathbf{x}\rightarrow\nu$ from the set of input patterns $\mathbf{x}$ into the set of output classes $\nu=1,\ldots,L$. In Bayesian terms, there is no input $\mathbf{x}$ for which the distributions or densities $p(\mathbf{x}|\lambda)$ corresponding to two or more different classes λ are simultaneously nonzero. Clearly, $P(\lambda|\mathbf{x})$ should reduce to $\delta_{\lambda,\nu(\mathbf{x})}$. The problem studied in the second half of the paper, discrimination between stable and unstable nuclides based on specified proton and neutron numbers, is actually of this type.

On the other hand, a *probabilistic* classification problem is one in which additional information beyond $\mathbf{x}$, not provided or unattainable in principle or in practice, is required to determine the class with certainty. In this situation, there does not exist a mapping $\nu(\mathbf{x})$ from input patterns to output classes, and $p(\mathbf{x}|\lambda)$ can be nonzero at the same $\mathbf{x}$ for different λ. (No commitment need be made regarding the character of the additional variables that influence the proper class assignment in a given occurrence of the pattern $\mathbf{x}$. These may or may not be random variables. The problem may be probabilistic without being stochastic [27].) The task of predicting protein secondary structure addressed in Refs. [28–30,3] is of this kind. Although in principle a full knowledge of the sequence of amino acid residues should determine the secondary-structure configuration (α helix, β sheet, or coil) in which each residue participates, the prediction is to be made on the basis of the portion of the sequence within a restricted window surrounding the target residue. In each such prediction, the portion of the sequence outside the window is effectively unknown, although it is perfectly definite. (Of course, the working assumption in such studies is that the unseen part of the sequence has little influence on the secondary-structure assignment.)

The clean distinction of problem types drawn above becomes muddy when one is confronted with typical real-world problems, where (i) in the deterministic case, the function $\nu(\mathbf{x})$, although existing in principle, is unknown, (ii) in the more general case of probabilistic classification, the conditional densities or probabilities $p(\mathbf{x}|\lambda)$, as well as the priors $P(\lambda)$, are unknown.

The Bayesian prescription itself loses some of its luster, since the ingredients on the right-hand side of Bayes' rule (1) must be *estimated*. (An additional real-world complication is that some of the inputs x_k supplied to the classifier may be irrelevant for the classification decision, or redundant.)

Whether one uses neural-network modeling or some more conventional method such as Parzen windows [2], the estimation process is normally based on a finite training sample (a set of assignments of training patterns to their corresponding classes). One imagines that the *a priori* distribution $P(\mu)$ has been sampled to generate a particular class λ, and then the class-conditional density or distribution $P(\mathbf{x}|\lambda)$ has been sampled to produce an input $\mathbf{x}$. In the case of deterministic classification, the stated aim is usually to estimate, or approximate, the mapping $\nu(\mathbf{x})$ rather than the *a posteriori* probability, and again the process of approximation of the target quantity involves extraction of the relevant information from a finite set of training samples. In general, limitations of both the estimation method and the training set will preclude complete precision in the reproduction of the performance of the ideal Bayes classifier or in the reconstruction of the deterministic map. In a neural-network approach, shortcomings of the estimation method may stem from an inadequate architecture or an inadequate training algorithm. In the next section, we introduce a type of neural network which, in principle, overcomes the architectural limitation.

III. HIGHER-ORDER PROBABILISTIC PERCEPTRONS

Definition. A higher-order probabilistic perceptron is a two-layer feed-forward neural network characterized as follows.

(1) The input layer consists of K units (labeled $k=1,\ldots,K$) whose activities $x_1,\ldots,x_K$ register the components of a given pattern vector $\mathbf{x}$.

(2) The output layer contains L units (labeled $\lambda=1,\cdots,L$ in a 1-1 correspondence with the pattern classes) whose activities $y_1(\mathbf{x}),\ldots,y_L(\mathbf{x})$ are to be computed from

$$y_\lambda(\mathbf{x})=\frac{e^{u_\lambda(\mathbf{x})}}{\sum_{\mu=1}^{L} e^{u_\mu(\mathbf{x})}}, \tag{2}$$

where u_λ is the net stimulus to the output unit λ from the units of the input layer. With the ''soft-maximum'' activation or ''squashing'' function (2), the output activities $y_\lambda(\mathbf{x})$ constitute a probability distribution over possible outcomes λ—they lie on the interval $[0,1]$ and sum to unity. This choice of squashing function has been employed by a number of authors (see, e.g., Refs. [3,10,22]).

(3) Each output unit λ receives synaptic connections, of strength $w_{\lambda,k_1\cdots k_m}$, from all distinct combinations $k_1\cdots k_m$ of inputs, with the order m running from $m=1$ to $m=K$. Accordingly, the stimulus u_λ appearing in Eq. (1) assumes the form

$$u_\lambda(\mathbf{x})=w_{\lambda,0}+\sum_{i=1}^{K} w_{\lambda,i}x_i+\sum_{i<j=1}^{K} w_{\lambda,ij}x_ix_j + \sum_{i<j<k=1}^{K} w_{\lambda,ijk}x_ix_jx_k+\cdots+w_{\lambda,12\ldots K}x_1x_2\cdots x_K, \tag{3}$$

where we have added a bias term $w_{\lambda,0}$. The indices i, j, and k are summed over the range $1,\ldots,K$, subject to the indicated restrictions. Note that a given coupling $w_{\lambda,k_1\ldots k_m}$ does not come into effect unless *all* of the transmitting units $k_1,\ldots,k_m$ have nonzero activities. (In the case of binary input patterns, all the transmitting units must be ''on.'') The bias $w_{\lambda,0}$ may be viewed as measuring the strength of a connection to unit λ from an effective external field and treated as a coupling of order $m=0$. The number of bias and weight parameters associated with each input λ is thus $\sum_{m=0}^{K}\binom{K}{m}=2^K$.

The HOPP system is a *perceptron* since it is a layered arrangement of neuronlike units with feed-forward connections from one layer to the next. Activation of the input units in a given pattern produces a set of output activities representing a ''percept'' associated with that pattern. More especially, the pattern may be assigned to a particular class λ by invoking a ''winner-take-all'' rule—the network is considered to have chosen class λ if the corresponding output unit is the most strongly active. The HOPP system is called *probabilistic* because specification (2) for the squashing function of the output units allows their activities to be interpreted as a probability distribution over the L classes λ. It is a *higher-order* network because the assumed two-layer architecture, specified by Eq. (3), incorporates not only pairwise connections from input to output units (as in the elementary perceptron [31,1]), but all higher-order connections consistent with the given number of inputs. The generic system having L output units may be viewed as a set of L single-output HOPP network elements that operate independently of one another—*except* as constrained by the condition $\Sigma_\lambda y_\lambda(\mathbf{x})=1$ imposed by the form (2) for the activation function.

IV. THE BAYES CONNECTION

We now develop a structural relationship between HOPP networks and Bayesian classifiers, specializing to the case of *binary input patterns* having $x_k\in\{0,1\}$ for $k=1,\ldots,K$.

Theorem. With a suitable choice of its synaptic couplings $w_{\lambda,k_1\cdots k_m}$ of all orders from $m=0$ to $m=K$, a higher-order probabilistic perceptron can reproduce the *a posteriori* probabilities provided by the Bayes optimal classifier. As a corollary, the minimum probability of misclassification (minimum error rate) is then achieved by imposing a winner-take-all decision at the output layer.

The validity of this theorem rests on (i) Bayes' rule of inference, (ii) a product decomposition of the class-conditional probabilities entering Bayes' rule, and (iii) the simple act of identifying, for each λ, the output activity $y_\lambda(\mathbf{x})$ of the HOPP with the Bayes *a posteriori* probability $P(\lambda|\mathbf{x})$ that pattern $\mathbf{x}$ belongs to class λ.

Proposition. Take $K=3$ for simplicity of expression. It is proposed that the class-conditional probability admits a product decomposition of the form

$$p(\mathbf{x}|\lambda)=\rho_1(x_1|\lambda)\rho_1(x_2|\lambda)\rho_1(x_3|\lambda)\rho_2(x_1x_2|\lambda) \times\rho_2(x_1x_3|\lambda)\rho_2(x_2x_3|\lambda)\rho_3(x_1x_2x_3|\lambda), \tag{4}$$

where all factors are positive semidefinite and together preserve $0\leq p(\mathbf{x}|\lambda)\leq 1$. This form has an obvious extension to arbitrary integral K. Specifically, a factor $\rho_m(x_{k_1}\cdots x_{k_m}|\lambda)$ is present for every distinct combination of m arguments $x_{k_1},\ldots,x_{k_m}$ taken from the set $\{x_1,\ldots,x_K\}$, with m running from 1 to K. The order chosen for the arguments $x_{k_1},\ldots,x_{k_m}$ is irrelevant, but the $K!/[m!(K-m)!]$ factors for given m may be different functions of their arguments. (For the sake of an uncluttered notation, these different factors will be distinguished only by the labels on their arguments.)

In practice such a decomposition (for arbitrary integral K) may be regarded as an identity. Its utility then hinges on the complexity of the decision tree of the problem at hand—on the possibility of accurate representation of $p(\mathbf{x}|\lambda)$ by a relatively small number of factors in the general product. If all factors except the first three on the right-hand side of Eq. (4) are approximated by unity, one has the familiar hypothesis [1] that all input components are independent. The independence hypothesis, which defines the *naive Bayes classifier*, is often made in connecting feedforward neural nets with Bayesian inference [1–3].

The product decomposition of the class-conditional probability $p(\mathbf{x}|\lambda)$ has a prominent counterpart in the structure of wave functions in quantum many-body physics. In particular, the ground state of a collection of indistinguishable bosons has an exact Feenberg product representation [32,33] in terms of one-body, two-body, three-body, . . . , factors reflecting the influence of a mean field and the existence of a

hierarchy of interparticle correlations. The structural analogy with Eq. (4) and its extensions to higher K is conceptually useful in spite of the fact that the wave function for the Bose system is symmetrical in all particle coordinates, whereas in the HOPP system the output units λ play a special role as receptors of stimuli and the input units are distinguishable from one another.

Following the lead of variational approaches to quantum many-body problems [35], it is natural to investigate the convergence of successive approximants to $p(\mathbf{x}|\lambda)$ in which higher-order factors $\rho_{m>M}(x_1 \cdots x_m|\lambda)$ are set equal to unity, for $M=1,2,\ldots$. The decomposition exemplified by Eq. (4) is more general than the product representation used by Duda and Hart [2] to generate the Chow expansion [36]. Curiously, the Duda-Hart representation has a structure analogous to that of the wave function used to model fermion pairing in superconductors [34].

Pursuing the analogy with correlated many-body wave functions somewhat farther, the hidden units in multilayer neural networks with only single-unit biases $w_{\lambda,0}$ and pairwise or ''two-body'' couplings $w_{\lambda,i}$ may be interpreted as the neural-network counterparts of the auxiliary ''shadow particle'' variables in shadow wave functions [37], in which real particles, shadows, and particles and shadows are correlated with each other exclusively via two-body correlation factors. The auxiliary shadow variables in such many-body wave functions mimic, to some degree, the existing higher-body correlations between the real particles. In a similar vein, the pairwise connections to and from the hidden units in multilayer networks take over tasks performed by higher-order connections in HOPP architectures.

It will be helpful to adopt a special notation for the values taken by the ρ factors in Eq. (4):

$$\rho_1(x_1=1|\lambda)=\rho_{1,\lambda}, \quad \rho_1(x_1=0|\lambda)=\rho_{\bar{1},\lambda},$$

$$\rho_2(x_1=1,x_2=1|\lambda)=\rho_{12,\lambda}, \quad \rho_2(x_1=0,x_2=1|\lambda)=\rho_{\bar{1}2,\lambda},$$

$$\rho_2(x_1=1,x_2=0|\lambda)=\rho_{1\bar{2},\lambda}, \quad \rho_2(x_1=0,x_2=0|\lambda)=\rho_{\bar{1}\bar{2},\lambda},$$

and similarly for higher-order factors, e.g.,

$$\rho_3(x_1=1,x_2=0,x_3=1|\lambda)=\rho_{1\bar{2}3,\lambda},$$

$$\rho_3(x_1=0,x_2=1,x_3=0|\lambda)=\rho_{\bar{1}2\bar{3},\lambda}.$$

Then we may, for example, express $\rho_3(x_1x_2x_3|\lambda)$ as

$$\begin{aligned}\rho_3(x_1x_2x_3|\lambda)=&\rho_{123,\lambda}^{x_1x_2x_3}\rho_{\bar{1}23,\lambda}^{x_1x_2(1-x_3)}\rho_{1\bar{2}3,\lambda}^{x_1(1-x_2)x_3}\\&\times\rho_{\bar{1}\bar{2}3,\lambda}^{(1-x_1)x_2x_3}\rho_{\bar{1}2\bar{3},\lambda}^{x_1(1-x_2)(1-x_3)}\\&\times\rho_{\bar{1}2\bar{3},\lambda}^{(1-x_1)x_2(1-x_3)}\rho_{\bar{1}\bar{2}3,\lambda}^{(1-x_1)(1-x_2)x_3}\\&\times\rho_{\bar{1}\bar{2}\bar{3},\lambda}^{(1-x_1)(1-x_2)(1-x_3)}.\end{aligned} \tag{5}$$

Identification. The crucial step in forging the stated connection between neural networks and Bayes inference is to identify the Bayes *a posteriori* probability for each class λ with the activity y_λ of the neural-network output unit assigned to that class:

$$P(\lambda|\mathbf{x})=\frac{p(\mathbf{x}|\lambda)P(\lambda)}{\sum_{\mu=1}^{L}p(\mathbf{x}|\mu)P(\mu)}=y_\lambda(\mathbf{x})=\frac{e^{u_\lambda(\mathbf{x})}}{\sum_{\mu=1}^{L}e^{u_\mu(\mathbf{x})}}. \tag{6}$$

We make the further identification

$$u_\lambda(\mathbf{x})=\ln[p(\mathbf{x}|\lambda)P(\lambda)]=\ln p(\mathbf{x}|\lambda)+\ln P(\lambda) \tag{7}$$

and calculate $\ln p(\mathbf{x}|\lambda)$ from the product expansion (4):

$$\begin{aligned}\ln p(\mathbf{x}|\lambda)=&\ln\rho_1(x_1|\lambda)+\ln\rho_1(x_2|\lambda)+\ln\rho_1(x_3|\lambda)\\&+\ln\rho_2(x_1x_2|\lambda)+\ln\rho_2(x_1x_3|\lambda)+\ln\rho_2(x_2x_3|\lambda)\\&+\ln\rho_3(x_1x_2x_3|\lambda).\end{aligned} \tag{8}$$

After some routine algebra based on the above definitions and representations such as Eq. (5), the result for $u_\lambda(\mathbf{x})$ can be cast in the advertised form (3) specific to $K=3$. Thus, the stimulus u_λ is composed of a bias of output unit λ plus singlet, doublet, and triplet stimuli to unit λ from the units of the input layer (i.e., feedforward couplings of orders 0, 1, 2, and 3). Extension of the Bayes connection to arbitrary K is a straightforward exercise.

V. EXPLICIT EXPRESSIONS

The explicit expressions for the biases $w_{\lambda,0}$ and couplings $w_{\lambda,i}$, $w_{\lambda,ij}$, $w_{\lambda,ijk}$, etc., are rather complicated, reflecting the combinatoric explosion of terms for growing K. However, a very interesting feature emerges: the bias or interaction $w_{\lambda,k_1k_2\ldots k_m}$ of order $m=0,1,2,\ldots,K$ contains a contribution from every order q in the product expansion with $m\leqslant q\leqslant K$, i.e., from all the $\rho_q(\cdots|\lambda)$ factors having q in the indicated range. This feature is expressed in the expansions

$$w_{\lambda,0}=\ln P(\lambda)+\sum_{q=1}^{K}w_{\lambda,0}^{(q)}, \quad w_{\lambda,i}=\sum_{q=1}^{K}w_{\lambda,i}^{(q)},$$

$$w_{\lambda,ij}=\sum_{q=2}^{K}w_{\lambda,ij}^{(q)}, \quad w_{\lambda,ijk}=\sum_{q=3}^{K}w_{\lambda,ijk}^{(q)}, \quad \text{etc.} \tag{9}$$

Complete to order $q=3$ in the product expansion of $p(\mathbf{x}|\lambda)$, we find

$$w_{\lambda,0}^{(0)}=\ln P(\lambda), \quad w_{\lambda,0}^{(1)}=\sum_{i=1}^{K}\ln\rho_{\bar{i},\lambda},$$

$$w_{\lambda,0}^{(2)}=\sum_{i<j=1}^{K}\ln\rho_{\overline{ij},\lambda}, \quad w_{\lambda,0}^{(3)}=\sum_{i<j<k=1}^{K}\ln\rho_{\overline{ijk},\lambda},$$

$$w_{\lambda,i}^{(1)}=\ln\left(\frac{\rho_{i,\lambda}}{\rho_{\bar{i},\lambda}}\right), \quad w_{\lambda,i}^{(2)}=\sum_{j=1}^{K}\ln\left(\frac{\rho_{i\bar{j},\lambda}}{\rho_{\bar{i}j,\lambda}}\right),$$

$$w_{\lambda,i}^{(3)}=\sum_{j<k=1}^{K}\ln\left(\frac{\rho_{ij\bar{k},\lambda}}{\rho_{\bar{i}\bar{j}k,\lambda}}\right),$$

$$w^{(2)}_{\lambda,ij}=\ln\left(\frac{\rho_{ij,\lambda}\rho_{\overline{ij},\lambda}}{\rho_{i\overline{j},\lambda}\rho_{\overline{i}j,\lambda}}\right),\quad w^{(3)}_{\lambda,ij}=\sum_{k=1}^{K}\ln\left(\frac{\rho_{ij\overline{k},\lambda}\rho_{\overline{ij}k,\lambda}}{\rho_{i\overline{j}k,\lambda}\rho_{\overline{i}j\overline{k},\lambda}}\right),$$

$$w^{(3)}_{ijk,\lambda}=\ln\left(\frac{\rho_{ijk,\lambda}\rho_{i\overline{jk},\lambda}\rho_{\overline{ij}\overline{k},\lambda}\rho_{\overline{i}j\overline{k},\lambda}}{\rho_{\overline{ijk},\lambda}\rho_{ij\overline{k},\lambda}\rho_{i\overline{j}k,\lambda}\rho_{\overline{i}jk,\lambda}}\right). \tag{10}$$

These formulas entail the use of definitions of the kind

$$\rho_{21}=\rho_{12},\quad \rho_{32}=\rho_{23},\quad \rho_{31}=\rho_{13},\quad \rho_{\overline{21}}=\rho_{\overline{12}},$$

$$\rho_{\overline{32}}=\rho_{\overline{23}},\quad \rho_{\overline{31}}=\rho_{\overline{13}},$$

$$\rho_{2\overline{1}}=\rho_{\overline{12}},\quad \rho_{3\overline{2}}=\rho_{\overline{23}},\quad \rho_{3\overline{1}}=\rho_{\overline{13}},\quad \rho_{1\overline{23}}=\rho_{13\overline{2}},$$

$$\rho_{\overline{123}}=\rho_{2\overline{13}},\quad \text{etc.},$$

and the convention that any $\rho\ldots$ with coincident indices (ignoring bar tags) vanishes. It is permissible to interchange two indices on a $\rho\ldots$ without disturbing the association of bars with indices, so as to obtain another name for the same quantity that is more convenient for presenting the results of the analysis. In deriving the above formulas we have not made specific use of the property that $\rho_2(x_1x_2|\lambda)$ should not be decomposable into a product of two independent one-input factors, and other similar restrictions.

The result (9) collecting higher-order contributions to lower-order couplings has a potentially important consequence for comparisons of feedforward neural networks and naive Bayesian classifiers. Structurally, the naive Bayes classifier, predicated on the independence hypothesis, is equivalent to the elementary perceptron [1], which has only input and output layers and output units with only biases $w_{\lambda,0}$ and pairwise incoming connections with weights $w_{\lambda,i}$. Yet the latter system may actually be superior in practice when trained on the real—correlated—data. The point is that the trained biases and pairwise interactions will in general include some effects of the higher-order correlations, i.e., they will incorporate some contributions from the factors $\rho_2,\rho_3,\ldots,\rho_K$ in the product expansion of the class-conditional probability. This feature is also present in the analogous treatment of Stolorz *et al.* [3] based on the Bahadur expansion.

VI. FORMAL AND PRACTICAL REMARKS

In this section we explore briefly some of the implications of our results and related findings.

Remark 1. We envision the following operation of the two-layer HOPP as an inference machine: in a training phase, the HOPP learns the statistics of the problem, which determine its weight parameters; in the computational phase, it effectively applies Bayes' rule (1).

Remark 2. The couplings and biases of a HOPP network may be determined by a training scheme that minimizes a well-chosen objective (or ''cost'') function. Reasonable choices of objective function include the squared error and the Kullback-Leibler distance or ''relative entropy'' [38], both computed over a set of training patterns. There exist strong results relating the computational output of suitably trained neural-network classifiers—notably, perceptrons—to the estimation of conditional probabilities, and indeed to the estimation of Bayesian posterior probabilities. These results have been reviewed and augmented by Richard and Lippmann [11]. The primary focus is the standard classification problem defined in Sec. II, which the latter authors would call a ''1 of L'' problem (meaning that the target output associated with the correct class is unity and all other target outputs are zero). For this case, Ruck *et al.* [9] have proven (see also Ref. [10]) that in the limit of an infinite, unbiased training sample, minimization of the mean-square deviation of actual output activities from their targets (over output units and patterns) also serves to minimize a mean-square-error measure of the departure of these output activities from the corresponding Bayes optimal discriminant functions. An alternative statement [11] is that under the same assumption, when network parameters are chosen to minimize a squared-error cost function, the outputs of the trained network provide direct estimates of the posterior probabilities of the Bayes classifier so as to minimize the mean-squared estimation error. Moreover, for the more general classification problem in which the desired outputs are binary and not necessarily ''1 of L'' [11], the actual outputs of the trained network estimate the conditional expectations of the desired outputs so as to minimize the mean-squared estimation error. (It should be pointed out that the proofs given in Refs. [9–11], unlike the demonstration of the Bayes structural equivalence for HOPP systems presented in Sec. IV, do not require that the input patterns $\mathbf{x}$ have binary components.)

Corresponding results hold when the cross-entropy objective function is employed. The argument given in Ref. [11] is readily adapted to establish the same properties for the Kullback-Leibler distance (also called the relative entropy).

Remark 3. In the context of the HOPP architecture, the approximation to the Bayes ideal that is at the heart of the results summarized in remark 2 can in principle be arbitrarily good, since the theorem of Sec. IV shows that the assumed hierarchy of couplings furnishes sufficient structural complexity to reproduce the Bayes recipe. However, one should be aware of a number of practical complications (as well as other caveats aired by Barnard [15]).

(a) It is not clear how one can actually attain the global minimum of the mean-square error if there are many local minima of the error surface.

(b) The requirement of an infinite, unbiased training sample is a strong one, and the quantitative consequences of deviations from this ideal need to be investigated.

(c) The implied minimization of the mean-square deviation of perceptron outputs from the corresponding Bayes optimal discriminant functions may not be sufficiently incisive. Consider the two-class problem for the case of continuous inputs $\mathbf{x}\in\mathcal{X}$, in the formulation of Ref. [9] where a perceptron with a single output $F(\mathbf{x})$ is trained to produce $+1$ when the input is from class 1 and -1 when it is from class 2. The mean-square error measure takes the form

$$\int_{\mathcal{X}}[F(\mathbf{x};w)-g_0(\mathbf{x})]^2p(\mathbf{x})d\mathbf{x}, \tag{11}$$

where $g_0(\mathbf{x})=P(\lambda=1|\mathbf{x})-P(\lambda=2|\mathbf{x})$ is the Bayes optimal discriminant and the squared deviation is appropriately weighted with the probability density $p(\mathbf{x})$ of the input vector. The output of the network will most closely approximate the Bayes discriminant function where $p(\mathbf{x})$ is large. Yet if

the aim is to minimize the probability of misclassification, the fit should be best on the decision boundaries of the classifier, where $g_0(\mathbf{x})=0$ (two-class problem) and $P(\lambda|\mathbf{x})=P(\mu|\mathbf{x})$, $\lambda,\mu=1,\ldots,L$ (general case). As demonstrated by Ruck *et al.* [9], these conditions do not generally occur where $p(\mathbf{x})$ is large.

Remark 4. Higher-order perceptrons enjoy universal computational properties quite apart from the Bayesian probabilistic view generally adopted in this paper. In particular, the HOPP system defined in Sec. III is a perfect generalization instrument for deterministic classification problems involving K binary inputs and L output classes. In this case, the $L\times 2^K$ weights associated with the HOPP architecture can be chosen to match an arbitrary Boolean target function.

Moving beyond pattern classification tasks, it is also evident that the basic HOPP architecture—an input layer coupled to an output layer via connections of arbitrary order consistent with the number of inputs—should have great utility in the more general class of problems considered in function approximation or regression (prediction of real variables) [39]. Target outputs as well as inputs and actual outputs may be continuous as well as discrete, and the problem may be either ''deterministic'' or ''probabilistic'' according to transparent extensions of the meanings assigned to these terms within pattern classification. Again, the system has the ability to capture correlations between the input variables of all orders, although a probabilistic interpretation of output activities (and, correspondingly, the adoption of softmaximum output functions) is in general inappropriate.

We note that Carmesin [40] has introduced ''multilinear'' neural networks containing, by definition, couplings of all orders from N input neurons to N output neurons. In contrast to the familiar case of perceptrons with one or more hidden layers (but only pairwise couplings), a convergence theorem for the back-propagation learning algorithm can be proven for this system, in which case back-propagation reduces to the so-called ''delta'' rule of Widrow and Hoff [41,23]. Given any input-output mapping, there exists a multilinear network that reproduces it, and back-propagation training produces convergence to such a network in a finite number of steps [40].

Remark 5. There is the obvious danger, in structures involving higher-order interactions, that the combinatorial explosion of weight parameters will overwhelm available computational resources and preclude tractable application even for relatively modest numbers K of input units. Two comments may be made to put this danger in perspective.

(a) If many higher-order correlations among the input variables are indeed present in the problem under consideration, the difficulty is intrinsic and must also be faced when attacking the problem using multilayer, pairwise-coupled feedforward networks. Accurate modeling will then generally demand large numbers of hidden units, optimally arranged in a single or multiple hidden layers. The HOPP scheme has the virtue of making all higher-order correlations explicit.

(b) Even though the number of input variables may be large in a given real-world problem, it will often be the case that only a few higher-order correlations are important, so that most of the weight parameters in the general HOPP architecture are zero or negligible. A knowledge of the problem domain may permit identification and elimination of these parameters. In Sec. VIII we shall propose and test a systematic procedure that may serve to reduce the parametric costs of HOPP networks to manageable proportions.

Remark 6. Another thrust in the articulation of Bayesian probability theory with neural networks, complementary to that explored here, involves the inference of the adaptive weight parameters or, more generally, treatment of the search in model space as an inference problem, based in any event on the given training data [13,14]. In Bayesian backpropagation [13], approximate Bayesian methods are applied to the determination of such statistical components of backpropagation as choosing a cost function and penalty term (or regularizer), pruning unimportant weights, estimating uncertainties in the weight parameters, predicting ''out-of-sample'' patterns, estimating generalization error, and comparing different network structures. A salient benefit of Bayesian model comparison is that it naturally incorporates Occam's razor [14]. Exploitation of this set of approaches may enhance the performance of higher-order (or ''multilinear'') networks.

VII. TRAINING ALGORITHMS

In preparation for a numerical demonstration of the capabilities of the HOPP networks as classifiers, we now develop supervised learning algorithms based on gradient-descent optimization of mean-square-error and relative-entropy objective functions. The development is motivated by the theoretical results [9–11] discussed in remark 2 of Sec. VI, and especially by the expectation that, in practice, HOPP networks trained by such algorithms will provide good estimates of Bayes *a posteriori* probabilities. The derivations proceed in close analogy with the formulation of standard errorback-propagation training schemes [23–26].

Having defined the HOPP network so that the activities of the output units form a complete set of probabilities over an exhaustive set of outcomes, it is possible to reduce the number of output units by one: $L\rightarrow L-1\equiv L'$. (Implicitly or explicitly, this is frequently done in the literature [3,22,42], most commonly in the case where a single output unit is used to represent a graded choice between two alternatives.) Elimination of one of the output units may be achieved by suitably renormalizing the weights (and biases) that enter the expression (3) for the stimuli $u_\lambda(\mathbf{x})$ felt by the output units. To this end, we divide the numerator and denominator of Eq. (2) by $\exp[u_L(\mathbf{x})]$, bringing the activity $y_\lambda(\mathbf{x})$ of output unit λ for input pattern $\mathbf{x}$ into the form

$$y_\lambda(\mathbf{x})=\frac{\exp[u_\lambda(\mathbf{x})-u_L(\mathbf{x})]}{1+\sum_{\mu=1}^{L'}\exp[u_\mu(\mathbf{x})-u_L(\mathbf{x})]}, \tag{12}$$

where $L'=L-1$. The renormalized stimuli $u_\lambda(\mathbf{x})-u_L(\mathbf{x})$ are seen to retain the general form of Eq. (3), and the differences $w_{\lambda,0}-w_{L,0}$, $w_{\lambda,i}-w_{L,i}$, $w_{\lambda,ij}-w_{L,ij}$, $\ldots$, $w_{\lambda,12\cdots K}-w_{L,12\cdots K}$ with $1\leqslant\lambda\leqslant L'$ emerge as the only independently adjustable parameters of the HOPP network. The notation is simplified by redefining $w_{\lambda,0}-w_{L,0}$ as $w_{\lambda,0}$, $w_{\lambda,i}-w_{L,i}$ as $w_{\lambda,i}$, $w_{\lambda,ij}-w_{L,ij}$ as $w_{\lambda,ij}$, $\ldots$, and

$w_{\lambda,12\cdots K} - w_{L,12\cdots K}$ as $w_{\lambda,12\cdots K}$. The activities of output units 1 through $L' = L-1$ are then given by

$$y_\lambda(\mathbf{x}) = \frac{\exp[u_\lambda(\mathbf{x})]}{1+\sum_{\mu=1}^{L'} \exp[u_\mu(\mathbf{x})]}, \quad 1 \leq \lambda \leq L'. \tag{13}$$

The normalization condition for the probability distribution $\{y_\mu(\mathbf{x}) | 1 \leq \mu \leq L\}$ is invoked to determine the activity associated with the deleted output unit corresponding to the remaining category L:

$$y_L(\mathbf{x}) = 1 - \sum_{\mu=1}^{L'} y_\mu(\mathbf{x}). \tag{14}$$

In view of the proliferation of weights that may occur in a HOPP system, it is best to take advantage of the saving offered by the above renormalization scheme, which is rigorous and independent of problem domain. The number of weight and bias parameters is reduced by a factor $(L-1)/L$ or by a total of 2^K parameters. One would like to keep the number of parameters as small as possible for two reasons. First, the problem of determining optimal weights and biases becomes more tractable computationally; and second, it is well known that the predictive capabilities of a network model are enhanced if the training data can be fitted with fewer parameters. We now proceed with the derivation of HOPP learning rules based on the familiar strategy of stepwise minimization of the chosen objective function. The following derivations are actually cast in terms of on-line rather than batch updating—i.e., weights and biases are adjusted after each pattern presentation rather than after each pass through the training corpus (each ''epoch''). Both options have been tried on the problem considered in Sec. VIII, and the ''on-line'' scheme was found to yield better results.

Let $y_\lambda(\mathbf{x})$ denote the activity generated by the network at output node λ in response to input pattern $\mathbf{x}$, and let $t_\lambda(\mathbf{x})$ be the corresponding target activity. Then the *pattern-specific* contribution to the objective function is

$$E(\mathbf{x}) = s(\mathbf{x}) = \sum_{\lambda=1}^{L} t_\lambda(\mathbf{x}) \ln\left[\frac{t_\lambda(\mathbf{x})}{y_\lambda(\mathbf{x})}\right] \tag{15}$$

when the relative-entropy prescription [22,38,26] is adopted and

$$E(\mathbf{x}) = e(\mathbf{x}) = \frac{1}{2}\sum_{\lambda=1}^{L} [t_\lambda(\mathbf{x}) - y_\lambda(\mathbf{x})]^2 \tag{16}$$

for the mean-square error measure. The relative-entropy cost function itself, denoted simply by s, is of course formed by averaging Eq. (15) over all patterns $\mathbf{x}$ in the appropriate set, while the usual squared-error cost function e (with the conventional factor 1/2) is obtained in the same manner from Eq. (16). Making use of the normalization relation (14) and the analogous relation satisfied by the target activities $t_\lambda(\mathbf{x})$, the pattern-specific cost functions (15) and (16) become

$$s(\mathbf{x}) = \sum_{\lambda=1}^{L'} t_\lambda(\mathbf{x}) \ln\left[\frac{t_\lambda(\mathbf{x})}{y_\lambda(\mathbf{x})}\right] + t_L(\mathbf{x}) \ln\left[t_L(\mathbf{x})\left(1+\sum_{\lambda=1}^{L'} \exp[u_\lambda(\mathbf{x})]\right)\right] \tag{17}$$

and

$$e(\mathbf{x}) = \frac{1}{2}\left\{\sum_{\lambda=1}^{L'} [y_\lambda(\mathbf{x}) - t_\lambda(\mathbf{x})]^2 + \left(\sum_{\lambda=1}^{L'} [y_\lambda(\mathbf{x}) - t_\lambda(\mathbf{x})]\right)^2\right\}. \tag{18}$$

It is to be noted that since the summations run only up to $L' = L-1$, only independent HOPP parameters are present in these formulas. In the case of either objective function, the absolute minimum value of zero is reached if and only if $y_\lambda(\mathbf{x}) = t_\lambda(\mathbf{x})$ for all $\lambda \in \{1,2,\ldots,L\}$. A potentially important difference between the behavior of the two cost functions [24] is that the squared deviation (16) or (18) saturates at a finite value of unity in case one of the target activities $t_\lambda(\mathbf{x})$ is unity and the corresponding network response $y_\lambda(\mathbf{x})$ is zero (or vice versa), whereas the relative entropy (15) or (17) tends to infinity for such extreme mismatches.

The usual gradient-descent minimization technique is adopted to derive learning rules based on the mean-square-error and relative-entropy cost functions. Thus, the weights and biases are to be incremented, after each pattern presentation, by an amount

$$\Delta w_\lambda(\mathbf{x}) = -\epsilon \frac{\partial E(\mathbf{x})}{\partial w_\lambda}, \tag{19}$$

where $E(\mathbf{x})$ is given by Eq. (17) or (18) and ϵ is a positive learning rate, generally small compared to unity. For economy of notation, we use w_λ as a generic symbol for the weight of any connection, of any order, received by output unit λ, or the bias parameter of that unit. It is of course convenient to regard the bias of a given unit as just another weight parameter—the weight of a connection to that unit from a fictitious ''field'' node 0 with perpetual activity $x_0 = 1$. To suppress wild oscillations in weight space, we follow the usual practice of supplementing the rule (19) by a so-called momentum term

$$\Delta w_\lambda(\mathbf{x}) = -\epsilon \frac{\partial E(\mathbf{x})}{\partial w_\lambda} + \eta \Delta w_\lambda(\mathbf{x}-1), \tag{20}$$

the positive momentum parameter η also being taken less than unity. In this expression, $\Delta w_\lambda(\mathbf{x}-1)$ stands for the last weight change made before the current one. Evaluation of the required partial derivatives leads to learning algorithms of the form

$$\Delta w_\lambda(\mathbf{x}) = \epsilon \Delta_\lambda(\mathbf{x}) \frac{\partial u_\lambda(\mathbf{x})}{\partial w_\lambda} + \eta \Delta w_\lambda(\mathbf{x}-1), \tag{21}$$

where $\Delta_\lambda(\mathbf{x})$ is interpreted as an error signal and depends on the cost function assumed.

For the relative-entropy cost function, the pattern-specific error signal takes the very simple form

$$\Delta_\lambda(\mathbf{x}) = t_\lambda(\mathbf{x}) - y_\lambda(\mathbf{x}), \tag{22}$$

while in the case of the squared deviation (18) it becomes

$$\Delta_\lambda(\mathbf{x}) = y_\lambda(\mathbf{x}) \left\{ t_\lambda(\mathbf{x}) - y_\lambda(\mathbf{x}) + \sum_{\mu=1}^{L'} [t_\mu(\mathbf{x}) - y_\mu(\mathbf{x})] \times \left[1 - \sum_{\nu=1}^{L'} y_\nu(\mathbf{x}) - y_\mu(\mathbf{x}) \right] \right\}. \tag{23}$$

The linearity of the stimuli (3) in the adaptable weight parameters w_λ has the consequence that the partial derivatives of $u_\lambda(\mathbf{x})$ appearing in Eq. (21) do not depend on these weights, but only on the training pattern $\mathbf{x}$ currently impressed at the input interface of the network. In the case of the bias parameter $w_\lambda = w_{\lambda,0}$ of node λ, the partial derivative reduces to unity for any input pattern. Generally, the derivatives $\partial u_\lambda(\mathbf{x})/\partial w_\lambda$ are given by the products of the activities, in pattern $\mathbf{x}$, of the input units belonging to the one-unit, two-unit, three-unit, etc. clusters extending feed-forward connections, of the pertinent order, to output unit λ. Independence of the weights may be exploited to facilitate computation: prior to a training run, one may calculate once and for all the pattern-specific values of the activity products for the given set of training patterns and store these values in an array for use during the training process. We note that for binary inputs the derivatives can only assume the values zero and unity.

From the expressions (22) and (23) it is seen that, in both training algorithms considered here, the training process is quenched as the actual network outputs $y_\lambda(\mathbf{x})$ approach the desired values $t_\lambda(\mathbf{x})$. However, the algorithm based on the mean-square error may slow down or become stalemated [24] upon encountering cases where one of the $y_\lambda(\mathbf{x})$ is 0 but should be 1 (or is 1 and should be 0). On the other hand, in the relative-entropy learning scheme the penalty for such errors may be too severe [22].

VIII. HOPP CLASSIFIERS FOR THE NUCLEAR STABILITY-INSTABILITY DICHOTOMY

At the most fundamental level, the independent input variables characterizing atomic nuclei are the proton number Z and the neutron number N. It is currently impractical to construct global quantum-mechanical models of nuclidic properties as functions of Z and N based on rigorous implementation of quantum chromodynamics, and even semiphenomenological effective hadronic theories fall short of true quantitative description. Accordingly, artificial neural networks and other modern statistical methods offer interesting alternatives [18–22] to such fundamental physical approaches to the prediction of nuclear structure and dynamics.

The efficacy and practicality of the HOPP architecture, in conjunction with the learning rules developed in Sec. VII, have been tested on the problem of classifying nuclides according to the stability or instability of their ground states. As pointed out in Sec. II, this example is actually a deterministic classification problem, whereas the HOPP scheme was formulated within the more general setting of probabilistic pattern classification. In principle, the inputs in this example—Z and N—provide sufficient information for the stability or instability of the corresponding nuclide to be determined, and the *a posteriori* probability $P(\lambda|\mathbf{x})$ in this case should collapse to a Kronecker delta $\delta_{\lambda,\nu(\mathbf{x})}$, where $\nu(\mathbf{x})$ is the physical mapping that uniquely delivers a stability or instability assignment for given Z,N. The latter mapping has not yet been derived from fundamental considerations. It is accessible only as an empirical lookup table, and then only under the assumption that there are no stable nuclides beyond those presently known. As in any exploration of the capabilities of neural-network models in classification or function approximation, a primary aim is to test generalization performance. Accordingly, the standard protocol requires the reservation of a subset of the data base as a test set, restricting the data available for training. One should not expect the ideal *a posteriori* probability or the target mapping to be realized in practice, for three reasons (see remarks 3 and 5 of Sec. VI). First, although the HOPP architecture is flexible enough to represent an arbitrary posterior probability distribution or deterministic mapping, computational limitations will necessitate a reduction in the number of weight parameters employed and therefore in the connectivity of the model. Second, the training set may be too small and/or it may be subject to bias. This is expected to be the most serious obstacle in the present example. Third, the training procedures may not be adequate; specifically, the required minimization of the chosen objective function may not be achieved. In the face of these uncertainties, the deterministic problem we have selected at least has the virtue, in contrast to most real-world problems of probabilistic character, that the form of the posterior probability distribution is *known*. It is true that the nuclear stability-instability problem is a rather academic one from a pragmatic viewpoint (unlike, say, the problem of predicting ground-state spins of nuclei). On the other hand, its computational demands are relatively modest, and results [19–21] from earlier modeling studies with standard multilayer neural networks are on hand for comparison.

Most of the known nuclei (and presumably all of the unknown ones) are unstable. The data base for our computational study, obtained from the National Nuclear Data Center at Brookhaven, is comprised of a total of 1557 nuclides, of which 215 are considered to be stable and 1305 are demonstrably unstable with respect to electron capture, β^- decay, α emission, or other modes of decay. A test set of 312 nuclides was formed by randomly selecting 260 unstable and 52 stable nuclides from the total data base. The remaining 1245 nuclides constitute the training set. The distributions of training and test sets in the N-Z plane are shown in Figs. 1(a) and 1(b).

For this problem, it suffices to employ a single analog output unit, whose activity signals the decision made by the network model regarding the stability or instability of the input nuclide. Target activity values $t(\mathbf{x})$ of 1 and 0, respectively, are assigned to stable and to unstable nuclides; imposing a winner-take-all criterion, it is assumed that the network has classified the input nuclide as stable [unstable] if the activity $y(\mathbf{x})$ of the output unit is greater than [less than] 0.5.

In the case of a single-unit output layer, the error signal (23) derived for the mean-square-error cost function may be reduced to

$$\Delta(\mathbf{x}) = 2y(\mathbf{x})[1 - y(\mathbf{x})][t(\mathbf{x}) - y(\mathbf{x})]. \tag{24}$$

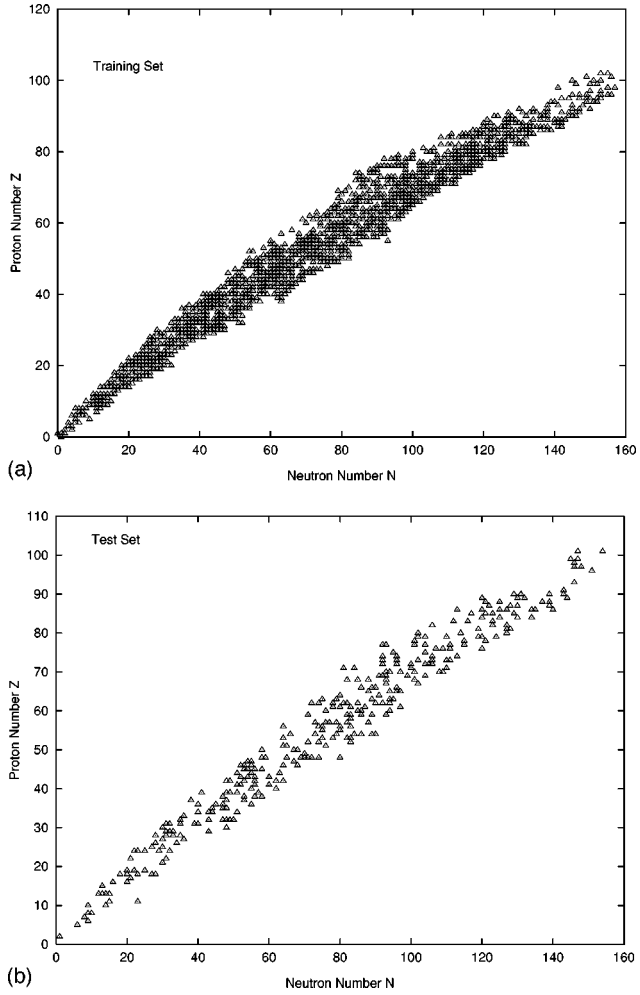

FIG. 1. (a) Training set of 1245 nuclides. (b) Test set of 312 nuclides.

Apart from an irrelevant factor 2, this result coincides with the usual back-propagation formula for the error signal generated by the output unit [23–26]. Just as in the familiar result, the product $y(\mathbf{x})[1-y(\mathbf{x})]$ is the derivative of the output-unit squashing function with respect to the stimulus $u(\mathbf{x})$ received by the output unit. In similar vein, it may be noted that the learning rule (21),(22) derived for the relative-entropy cost function has the form of the Widrow-Hoff [41] ''delta'' learning rule, extended to allow for higher-order connections and including a momentum term.

As in most applications of neural networks to the modeling of nuclear systematics [17–22], we adopt a binary representation of the inputs Z and N. Thus, the input layer of the HOPP network models to be studied consists of eight on-off units that encode Z as a bit string and another eight on-off units that similarly encode N. In contrast to analog coding of Z and N in the activities of one or more input nodes, this binary coding scheme emphasizes the integral (quantal) character of the input variables [19,20] and hence provides a more natural framework for modeling effects of shell structure and pairing [43], effects which can be crucial in distinguishing between stability and instability.

In addition to the squared error and relative entropy, averaged over training or test patterns, we have also used two other quality measures to assess network performance in learning and predictive modes. One is simply the number (or percentage) of correctly classified input patterns. However, this measure can easily be misleading because of the predominance of unstable examples in the training and test sets. A ''lazy'' net that routinely classifies *all* input nuclides as unstable would achieve scores of 83.94% correctly classified examples on the training set and 83.33% on the test set. However, such apparently strong performance would only indicate that the network has learned of the high likelihood that an arbitrarily selected nuclide is unstable.

A much more informative quality measure is the Mathews correlation coefficient [44]

$$C=\frac{p\overline{p}-q\overline{q}}{\sqrt{(p+q)(\overline{p}+q)(p+\overline{q})(\overline{p}+\overline{q})}}, \tag{25}$$

which is constructed to eliminate the effect of bias from first-order frequencies. In this formula, p is the number of stable input nuclides correctly classified as stable, $\overline{p}$ the number of unstable input nuclides correctly classified as unstable, q the number of stable input nuclides incorrectly classified as unstable, and $\overline{q}$ the number of unstable input nuclides incorrectly classified as stable. The value of the coefficient C ranges between -1 and $+1$, taking its minimum value of -1 when all patterns are misclassified and its maximum or ideal value $+1$ when all patterns are correctly classified. In the case of a network model that assigns all patterns to one class (e.g., unstable), C vanishes, indicating trivial performance.

The Mathews coefficient is computed independently for the training and test sets. Evidently, $p+q$ is just the total number of stables in the data set considered and $\overline{p}+\overline{q}$ is the total number of unstables, so the Mathews coefficient is in effect a function only of the number p of correctly identified stables and the number $\overline{p}$ of correctly identified unstables. Figure 2 shows this function $C(p,\overline{p})$ for the training set.

In our modeling experiments, each training run comprises a prescribed number of epochs of exposure to the training patterns. Since an on-line (or ''stochastic'') training procedure is adopted, all four performance measures considered—mean-square error measure e, pattern-averaged relative entropy s, number N_{corr} of correct classifications, and Mathews coefficient C_{train}—will show fluctuations throughout the training process. These quantities are monitored during each training run and the sets of weights yielding the minimum e, minimum s, maximum N_{corr}, and maximum C_{train} are recorded. Generally, the HOPP network corresponding to the maximum Mathews coefficient found during a given run displays predictive performance superior to that of the network configuration reached on completion of the run.

As specified above, the HOPP architecture for the stability discrimination problem consists of 16 input units and a single output node. Allowing for feed-forward connections of all orders, this structure entails $\Sigma_{m=0}^{16}\binom{16}{m}=2^{16}=65536$ adjustable weight parameters. Quite apart from the computational demands of training this huge set of weights, there is little point in such an exercise. Far more parametric resources are available than are needed to construct a lookup-table that recapitulates the content of the training set. Thus it can be expected that one may readily arrive at networks

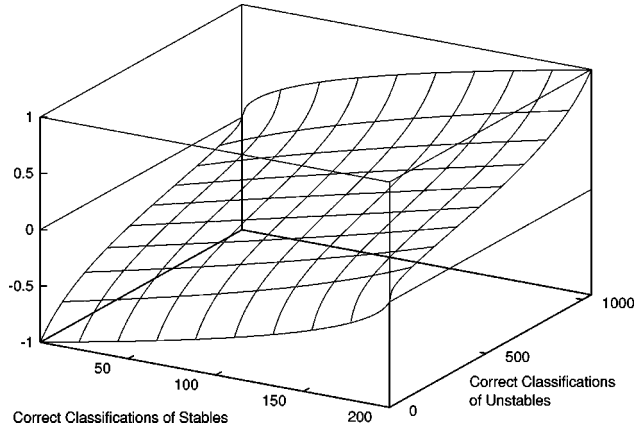

FIG. 2. Mathews correlation coefficient (25), for the training set, as a function of the number $p=0,1,2,\ldots,200$ of correctly identified stables and of the number $\bar{p}=0,1,2,\ldots,1045$ of correctly identified unstables.

which fit the training data perfectly, yet fail in generalizing to new patterns. This expectation is borne out by experiments in which the connectivity was truncated at order $m=3$, i.e., experiments on HOPP networks with 697 adjustable parameters. (Connections of order $m=3$ are called ''quaternary,'' since they involve four neurons—three input neurons and one output neuron.) In several cases, training led to a perfect fit of the training data, but performance on the test set was poor, with Mathews coefficients below 0.25.

Already at $K=16$, one feels the effect of the combinatoric explosion of parameters inherent in the HOPP architecture. Some means must be found to preselect relevant connections, reducing the number of adjustable parameters to a level that permits good generalization on the basis of the limited training set available. The following strategy is quite effective in the stability discrimination problem.

As was emphasized in Sec. III, the connection weight $w_{\lambda,k_1\cdots k_m}$ comes into play for a given input pattern if and only if the input units $k_1\cdots k_m$ are all ''on.'' In the case of bias parameters ($m=0$) this condition is of course trivially met for all training and test nuclides. More generally, for each of the 65 536 possible connections one must count the number of nuclides among the 1245 in the training set for which this connection is activated and therefore necessary. While time consuming, this task need be done only once for a given data set. The next step is to reduce the number of couplings considered, by retaining only those which become active for *at least* $N_{\rm crit}$ examples in the training set. In choosing the value of $N_{\rm crit}$, there is an inevitable tradeoff between accuracy on the training set (arguing for small $N_{\rm crit}$) on the one hand, and predictive accuracy and practicality of training on the other (arguing for larger $N_{\rm crit}$).

Table I shows the number of connections activated by at least one training example ($N_{\rm crit}=1$), in each order from $m=0$ to $m=16$. As expected, this number first increases rapidly with increasing order, reaches a maximum at intermediate order, and thereafter falls off very rapidly. The number of relevant parameters is immediately reduced to 12 435; however, this number is still too large in comparison with the available training set.

The best three HOPP ''inference engines'' we have found in our numerical explorations have $W=109$, 127, and 143 weights (including biases). These nets correspond, respectively, to values 200, 170, and 165 for $N_{\rm crit}$. Tables II–IV display, for each order m listed, the number of connections coming into action for at least $N_{\rm crit}=200$, $N_{\rm crit}=170$, and $N_{\rm crit}=165$ nuclides in the training set. For values of m higher than those listed, there are no connections excited by $N_{\rm crit}$ or more nuclides.

TABLE I. The number of connections for every order that come into effect for at least one nuclide in the training set ($N_{\rm crit}=1$). The first column gives the number m of input units feeding into the output unit, the second the number W_m of connections of order m coming into action for at least $N_{\rm crit}$ training examples, and the last the total number of possible connections of the order specified in the first column.

m	W_m	$\binom{16}{m}$
0	1	1
1	15	16
2	103	120
3	424	560
4	1161	1820
5	2204	4368
6	2932	8008
7	2751	11440
8	1807	12870
9	793	11440
10	211	8008
11	31	4368
12	2	1820
13	0	560
14	0	120
15	0	16
16	0	1
Total	12 435	65 536

It is worth noting that this procedure for sculpting the connectivity of HOPP networks permits one to include connections of *any* order that satisfy the relevancy criterion of activation by at least $N_{\rm crit}$ training examples. The parameter-reduction strategy we have introduced is likely to have wide applicability.

Training a HOPP network with the number W of relevant weights began with the assignment of an initial value of zero to all of these weights. (Nets have also been trained by starting with initial weights chosen randomly from a uniform distribution on $[-0.5,0.5]$, with no significant difference in the quality of the models derived.) Each training run consisted of $N_{\rm max}=36\,000$ epochs. Within an epoch, all nuclides of the training set were presented in random order, the order

TABLE II. Same as in Table I for $N_{\rm crit}=200$. The table is truncated at the order above which no more connections come into effect for at least $N_{\rm crit}$ training examples.

m	W_m	$\binom{16}{m}$
0	1	1
1	14	16
2	89	120
3	5	560
Total	109	697

TABLE III. Same as in Table II for $N_{crit}=170$.

m	W_m	$\binom{16}{m}$
0	1	1
1	14	16
2	90	120
3	22	560
Total	127	697

of presentation being changed (randomly) from epoch to epoch. Weights were updated upon each pattern presentation based on Eqs. (21) and (22) when training on the relative entropy and according to Eqs. (21) and (23) when training on the mean-square error. However, to speed up the training process these weight changes were implemented if and only if the HOPP response $y(\mathbf{x})$ to input pattern $\mathbf{x}$, evaluated with the current set of weights, departs from the corresponding target value $t(\mathbf{x})$ by an amount greater in magnitude than 0.5.

About a hundred HOPP networks have been trained in our numerical investigations, in which we have explored the effects of different choices for the relevancy cutoff N_{crit}, the learning rate ϵ, the momentum parameter η, and the maximum number N_{max} of epochs per training run. Table V summarizes the properties of the best eight of these network models. We have seen that an exact fit of the training data is not useful in itself; thus ''best'' is to be understood in terms of the best performance on the test set, consistent with good accuracy on the training set. These networks are identified by number in the first column of the table. The second column gives the number of connections that remain after applying the parameter reduction procedure described above, in correspondence with the choices of N_{crit} considered in Tables II–IV. The entries in the third column indicate whether the mean-square-error (MSE) or mean-relative-entropy (MRE) cost function (CF) was adopted. The corresponding learning rates ϵ and momentum parameters η are listed in the fourth and fifth columns, respectively.

The final configuration of any of the networks 1–8 was in fact reached through a succession of training runs of 36 000 epochs. For nets 1 and 4 [2 and 6], successive runs were started with the weights corresponding to the minimum value of the mean relative entropy [mean-square error] achieved in the immediately preceding run, while for nets 3, 5, 7, and 8 the initial weights for a subsequent run were taken as those corresponding to the maximal Mathews coefficient found in the most recent training run. In the seventh column, we list, for each network, the value of the Mathews coefficient C_{train}. The number of epochs N_{best} at which this value was obtained is given in column six, and the corresponding value

TABLE IV. Same as in Table II for $N_{crit}=165$.

m	W_m	$\binom{16}{m}$
0	1	1
1	14	16
2	90	120
3	38	560
Total	143	697

TABLE V. Performance of the eight best HOPP classifiers for the nuclear stability-instability discrimination problem, found in a large set of numerical simulations. The networks labeled 1–8, having W connections, were trained by modified on-line gradient-descent learning algorithms based on the mean-relative-entropy (MRE) or mean-square-error (MSE) cost function (CF), with learning rate ϵ and momentum parameter η. The Mathews correlation coefficient obtained by each net for the training set (C_{train}) is given along with the number of epochs at which this value was achieved (N_{best}) and the corresponding Mathews coefficient on the test set (C_{test}). See text for further details.

	W	CF	ϵ	η	N_{best}	C_{train}	C_{test}
1	143	MRE	0.025	0.2	224 769	0.79	0.63
2	109	MSE	0.025	0.9	142 982	0.68	0.63
3	127	MSE	0.025	0.5	229 718	0.74	0.63
4	143	MRE	0.025	0.5	170 254	0.80	0.63
5	143(133)	MSE	0.025	0.9	137 884	0.77	0.63
6	143	MSE	0.025	0.9	147 854	0.81	0.62
7	143	MRE	0.025	0.0	89 400	0.78	0.62
8	143	MSE	0.025	0.9	142 660	0.81	0.61

of the Mathews coefficient C_{test} for the test set is listed in the last column.

As already mentioned, the fully trained networks designated 1–8 have been singled out from all those studied by virtue of their superior predictive performance on the stability-instability discrimination problem. The weight configurations of networks 1 and 4, 2 and 6, and 5, correspond, respectively, to the maximal Mathews coefficient, the minimal mean relative entropy, and the minimal mean-square deviation found in the last run of 36 000 training epochs. Similarly, the patterns of weights in nets 3, 7, and 8 are those belonging to the maximal number of correctly identified training examples recorded in the final run. Further training of the HOPP systems of Table V did not significantly improve their performance.

Network 5 represents a special case in which an attempt was made to eliminate further unimportant weight parameters. The relevancy criterion based on the cutoff choice $N_{crit}=165$ yielded the same preselected HOPP architecture as for networks 4, 6, 7, and 8, all having 143 relevant connections (including biases). Training led to an intermediate network corresponding to the maximal Mathews correlation coefficient obtained in the first 108 000 training epochs (occurring at 100 894 epochs). This network was pruned by the following procedure. For each of the 143 connections, the Mathews coefficient for the training set was calculated with the strength of that connection set to zero, leaving the other weights unchanged. The connection whose omission resulted in the least deterioration in the Mathews coefficient was deleted. This process was in turn applied to the pruned network, and iterated until a total of ten connections were removed. The reduced importance of the deleted weights is evident in their small magnitudes as well as their effect on C_{train}. The resulting HOPP architecture, with 133 nonvanishing weights, provided the initial configuration for the remaining 36 000 epochs of the training regimen.

A similar pruning strategy has been employed extensively in neural-network modeling of the atomic mass table [21]

and the systematics of nuclear decay [22], where it resulted in significant improvement in the quality of the statistical models. In the present problem, further pruning of HOPP networks preoptimized using the relevancy criterion usually did not produce significantly better networks. Network 5 is one of the rare cases where there was a noticeable improvement in performance.

The nuclear stability-instability discrimination problem has been studied earlier using standard multilayer feedforward nets with pairwise connections [19–21]. Among a large number of models generated by training with gradient-descent back-propagation routines (batch as well as on-line) and with a conjugate gradient algorithm, the best performance was recorded by a network with a four-layer architecture $16+9+6+2$ and 227 weight parameters. Again, binary input coding was adopted; two analog output neurons coded for ''stability'' and ''instability,'' the discrimination decision being made by a winner-take-all rule. This network, developed by conjugate-gradient training, attained Mathews coefficients of $C_{\text{train}}=0.87$ and $C_{\text{test}}=0.68$. The results collected in Table V show that HOPP networks are capable of comparable performance, with substantially better parametric efficiency. For example, network 6 has 143 parameters compared to 227 for the best multilayer network model. It may also be noted that network 2, with only 109 weights, almost matches the predictive performance of network 1, as measured by C_{test}. However, the additional weight parameters possessed by network 1 result in significantly better learning accuracy.

The range of relevancy parameters from $N_{\text{crit}}=200$ to $N_{\text{crit}}=165$ was found to produce an acceptable tradeoff between accuracy of fit and predictive power. Within this range, where the number of connections runs from 109 to 143, well-trained HOPP models show remarkably little spread in C_{test} values, although the details of connectivity are quite variable from model to model. This robustness is consistent with the fact that there was generally little to be gained by attempts to improve upon the preselected connection pattern by the pruning routine. For relevancy-optimized architectures such as those of Tables II–IV it is not hard to construct HOPP networks with a predictive Mathews coefficient close to 0.6. However, it has proven difficult to go much beyond this value, either with the HOPP architectures adopted or with the more conventional multilayer feedforward architectures. This difficulty is a reflection of the complexity of the distribution of stable and unstable nuclides in the N-Z plane, where the stables intermingle with unstables in an intricate pattern along the valley of stability in a sea of instability. Indicative of this complexity are the very long training times needed to create the networks of Table V. It is of some interest to compare the present situation with that encountered in an even more difficult classification problem, namely, that of predicting protein secondary structure from the primary amino acid sequence [28–30,3,45,46]. It has been standard practice to treat this problem as one of probabilistic classification, supplying a multilayer, feedforward, pairwise-coupled neural network with partial sequence information passed through a sliding or jumping window. Such approaches—as well as other statistical methods based on sequence information alone—have not succeeded in yielding Mathews coefficients for alpha-helix prediction significantly beyond about 0.4 (see Refs. [3,30]).

It seems most likely that the principal obstacle to improved performance on the stability-instability discrimination problem is the limited size of the data set, which prevents one from maintaining predictive accuracy upon reduction of the relevancy cutoff N_{crit} below the range considered for Tables II–IV. In this connection, we note that the training procedures based on the relative-entropy and squared-error objective functions produced optimized networks of essentially equivalent quality. Indeed, in many cases it was found that, within a given training run in the late stages of development of the better networks, the weight configurations corresponding to the minimum mean relative entropy and the minimum mean-square error were reached after nearly the same number of completed epochs and are thus very close to one another. Such behavior, as well as the very similar performance of the best networks of HOPP and standard multilayer type, suggests that the existing solutions of the stability classification problem are about as good as can be expected, given the complexity of this problem and the restricted nature of the data set. Even so, one cannot rule out the existence of superior neural-network nuclear-stability classifiers of one kind or another, which might be accessible through the Bayesian techniques of model search developed in Refs. [13,14] (see remark 6 of Sec. VI).

IX. CONCLUSIONS

We have studied a perceptron architecture for solving classification problems that contains only input and output layers but (i) allows for synaptic connections of all orders consistent with the number of input units and (ii) incorporates output-unit activation functions that collectively compute a probability distribution over an exhaustive set of outcomes. Assuming binary (two-state) input units, it has been established that such higher-order probabilistic perceptrons (HOPPs) are sufficiently general to represent the correlations among the input variables and generate the *a posteriori* probabilities given by a Bayes-optimal classifier. In addition, we have developed supervised learning algorithms for training HOPP networks on a set of correctly classified examples, with the goal of achieving optimal or near-optimal performance. The introduction of these algorithms, which employ mean-square-error and relative-entropy objective functions, has its basis in theorems enunciated by Ruck *et al.* [9] and Richard and Lippmann [11].

The HOPP architecture and associated learning rules have been applied to the nontrivial task of deciding on the stability or instability of nuclear ground states. One putative disadvantage of HOPP networks is poor scaling with problem size, as measured by the dimensionality of the input space. If all possible connections are retained, there is a combinatoric explosion of higher-order weight parameters as the number of input variables is increased. We have sought to overcome this difficulty by imposing a relevancy criterion on the retention of weight parameters: all connections that are not excited by at least N_{crit} training patterns are deleted. This strategy, aimed at improving parametric efficiency while preserving important higher-order correlations, may prove

useful in a wide range of classification problems. It is found that relevancy-optimized HOPP solutions for the stability-instability discrimination task are competitive with solutions obtained with conventional multilayer perceptrons trained by back-propagation and conjugate gradient procedures.

It should be emphasized that conventional feedforward neural networks, with one or more hidden layers but only biases and pairwise connections, may also suffer from poor scaling behavior. The danger is merely more apparent in the case of HOPP networks, which *start* with a (finite) architecture general enough to match the correlation structure of any problem with the specified numbers of inputs and outputs. In practice, what really matters is the complexity of the problem at hand, in terms of the pattern of the higher-order correlations between input variables. In applying HOPP nets to large problems, the immediate need is to trim the architecture to suit the given problem. In the case of traditional nets, the existing complexity must be dealt with by introducing a sufficient number of hidden units, in one or more hidden layers, again with the prospect of an explosion of resource demands. Optimization of this architecture is in general a very hard problem. [Indeed, even an infinite number of hidden nodes may be required in a three-layer net (see Ref. [12]).] The HOPP architecture is an attractive alternative to that of traditional networks, in at least two respects.

(i) It provides a framework for efficient and systematic optimization strategies. A transparent option is to truncate the expansion (3) at successively higher orders until satisfactory performance is attained, within the limitations of the data set available. Optimization based on the relevancy criterion has the advantage that the most important links in expansion (3) may be selected regardless of order.

(ii) With input units directly connected to outputs, it provides for more straightforward (and more rapid) training of a given number of weight parameters.

On the other hand, the absence of a hidden layer may be construed to imply a sacrifice in parallel distributed processing, a feature of multilayer networks that is commonly regarded as highly advantageous [23]. Still, it must be remembered that both HOPP and conventional architectures (the latter with at least one hidden layer that can contain an arbitrary number of units) provide the raw material for universal machines. Whereas in HOPPs it is the presence of higher-order couplings that permit the description of all possible correlations between input patterns, in conventional approaches this responsibility falls on the set of hidden units. Formal relationships and correspondences between two-layer HOPPs and pairwise-coupled multilayer perceptrons are yet to be explored, as are more practical issues such as systematic differences in parametric efficiency and fault tolerance. For the illustrative problem studied in this paper, the existing simulations suggest that the two approaches are comparably effective for the primary task of generalization.

Even for problems of *modest* size, a large fraction of the possible weight parameters must be eliminated if training of HOPP models is to be practicable and if acceptable performance in generalization is to be achieved with available data sets. Thus, the viability of statistical modeling with HOPP networks depends on the convergence of proposed optimization strategies. Within this context, it is instructive to consider the correlation structure of the stability-instability discrimination problem in terms of the importance of connections of various orders. First, it should be recalled that the elementary perceptron with only biases and pairwise connections is inadequate, yielding Mathews correlation coefficients C_{test} on the test set of at best slightly larger than 0.3. Based on the relevancy criterion (as reflected in Tables II–IV) the training data indicate that the most important connections for deciding stability are the binary (i.e., pairwise) and ternary ones, the quaternary connections being far less important. Indeed, in our computer experiments, the great majority of the ternary connections had to be retained to obtain good (or ''optimized'') HOPP models. It is of course these ternary connections (along with a few quaternary links) which introduce the nonlinearities that are represented in the multilayer structure of the traditional architecture and permit C_{test} to be raised above 0.6. The situation is exemplified by comparisons drawn from the experiments summarized in Table V. With 33 fewer quaternary connections and one less ternary coupling, network 2 is almost as good in generalization as network 1. This illustrates the relative unimportance of quaternary connections, as does the fact that of the ten connections deemed least important when pruning network 5, only two are ternary while eight are quaternary.

Our formal and computational findings suggest that higher-order probabilistic perceptrons should be useful in many classification problems that arise in science and technology, whether probabilistic or deterministic in character. Among scientific applications, one may consider the prediction of diverse aspects of protein structure related to protein folding [45,46] and classification of compounds according to crystal structure, as well as further global modeling exercises in nuclear physics such as assignment of ground-state spins and parities to novel nuclear species. As a natural sequel to the present study of the stability-instability dichotomy, HOPP systems may be taught to generate a probability distribution over the possible fates of the ground state of an input nuclide, which may include stability and decay into various modes (α decay, β decay, electron capture, fission, . . .). The relevant body of data, referring to nearly 1600 nuclear ground states, consists of stability assignments or branching probabilities for the observed modes of decay. Although this problem is one of function approximation rather than simple classification, it is nevertheless amenable to HOPP modeling (see remark 4 of Sec. VI). Moreover, it is of considerable interest in view of the predictive success achieved with pairwise-coupled multilayer feedforward nets trained on the relative-entropy cost function [22]. The complexity of the problem is expected to remain within the bounds of practical computation, since the presence of (say) five possible decay modes implies a number of initial HOPP weights five times that of the stability-instability discrimination task.

ACKNOWLEDGMENTS

This research has been supported by the National Science Foundation under Grant No. PHY-9602127, and by the Deutcheforschungsgemeinschaft under Graduiertenkolleg GK 14 at the University of Cologne, and by the European Union Human Capital and Mobility Scheme.

[1] H. Minsky and S. Papert, *Perceptrons* (MIT Press, Cambridge, MA, 1969).
[2] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis* (Wiley, New York, 1973).
[3] P. Stolorz, A. Lapedes, and Y. Xia, J. Mol. Biol. **225**, 363 (1991).
[4] R. R. Bahadur, in *Studies in Item Analysis and Prediction*, edited by H. Solomon (Stanford University Press, Stanford, CA, 1961), p. 158.
[5] C. H. Anderson and E. Abrahams, in *Proceedings of the IEEE First International Conference on Neural Networks*, San Diego, 1987, edited by M. Caudill and C. Butler (IEEE, New York, 1987), Vol. 3, p. 105.
[6] H. White, Neural Comput. **1**, 425 (1989).
[7] A. Lansner and Ö. Ekeberg, Int. J. Neural Syst. **1**, 77 (1989).
[8] S. P. Luttrell, in *Maximum Entropy and Bayesian Methods*, edited by J. Skilling (Kluwer, Dordrecht, 1989), p. 363.
[9] D. W. Ruck, S. K. Rogers, M. Kabrisky, M. E. Oxley, and B. W. Suter, IEEE Trans. Neural Netw. **1**, 296 (1990).
[10] E. A. Wan, IEEE Trans. Neural Netw. **1**, 303 (1990).
[11] M. D. Richard and R. P. Lippmann, Neural Comput. **3**, 461 (1991).
[12] M. Opper and D. Haussler, Phys. Rev. Lett. **66**, 2677 (1991).
[13] W. L. Buntine and A. S. Weigend, Complex Syst. **5**, 603 (1991).
[14] D. J. C. MacKay, Neural Comput. **4**, 415 (1992); **4**, 448 (1992); **4**, 698 (1992).
[15] E. Barnard, IEEE Trans. Neural Netw. **3**, 1026 (1992).
[16] T. L. H. Watkin, A. Rau, and M. Biehl, Rev. Mod. Phys. **65**, 499 (1993).
[17] *Scientific Applications of Neural Nets*, edited by J. W. Clark, T. Lindenau, and M. L. Ristig (Springer-Verlag, Berlin, 1999), Vol. LNP 522.
[18] K. A. Gernoth and J. W. Clark, Comput. Phys. Commun. **88**, 1 (1995).
[19] S. Gazula, J. W. Clark, and H. Bohr, Nucl. Phys. **A540**, 1 (1992).
[20] J. W. Clark, S. Gazula, K. A. Gernoth, J. Hasenbein, J. S. Prater, and H. Bohr, in *Recent Progress in Many-Body Theories*, edited by T. L. Ainsworth, C. E. Campbell, B. E. Clements, and E. Krotscheck (Plenum Press, New York, 1992), Vol. 3, p. 371.
[21] K. A. Gernoth, J. W. Clark, J. S. Prater, and H. Bohr, Phys. Lett. B **300**, 1 (1993).
[22] K. A. Gernoth and J. W. Clark, Neural Networks **8**, 291 (1995).
[23] D. E. Rumelhart, J. L. McClelland, and the PDP Research Group, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* (MIT Press, Cambridge, MA, 1986), Vol. 1.
[24] J. Hertz, A. Krogh, and R. G. Palmer, *Introduction to the Theory of Neural Computation* (Addison-Wesley, Redwood City, CA, 1991).
[25] B. Müller, J. Reinhardt, and M. T. Strickland, *Neural Networks*, 2nd ed. (Springer-Verlag, Berlin, 1995).
[26] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd ed. (Prentice Hall, Upper Saddle River, NJ, 1999).
[27] E. T. Jaynes, in *Maximum-Entropy and Bayesian Methods in Applied Statistics*, edited by J. H. Justice (Cambridge University Press, Cambridge, 1986), p. 1.
[28] N. Qian and T. J. Sejnowski, J. Mol. Biol. **202**, 865 (1988).
[29] H. Bohr, J. Bohr, S. Brunak, R. M. J. Cotterill, B. Lautrup, L. Nøskov, O. H. Olsen, and S. B. Petersen, FEBS Lett. **241**, 223 (1988).
[30] D. G. Kneller, F. E. Cohen, and R. Langridge, J. Mol. Biol. **214**, 171 (1990).
[31] F. Rosenblatt, *Principles of Neurodynamics* (Spartan Books, Washington, DC, 1962).
[32] E. Feenberg, Ann. Phys. (N.Y.) **84**, 128 (1974).
[33] J. W. Clark, Nucl. Phys. **A328**, 587 (1979).
[34] J. R. Schrieffer, *Theory of Superconductivity* (Benjamin, New York, 1964).
[35] *Microscopic Quantum Many-Body Theories and Their Applications*, edited by J. Navarro and A. Polls (Springer-Verlag, Berlin, 1998), Vol. LNP 510.
[36] C. K. Chow and C. N. Liu, IEEE Trans. Inf. Theory **14**, 462 (1968).
[37] F. Pederiva, S. A. Vitiello, K. A. Gernoth, S. Fantoni, and L. Reatto, Phys. Rev. B **53**, 15 129 (1996).
[38] S. Kullback, *Information Theory and Statistics* (Wiley, New York, 1959).
[39] J. H. Friedman, in *From Statistics to Neural Networks. Theory and Pattern Recognition Applications*, edited by V. Cherkassky, J. H. Friedman, and W. Wechsler (Springer-Verlag, Berlin, 1994), p. 1.
[40] H.-O. Carmesin, Phys. Lett. A **188**, 27 (1994).
[41] G. Widrow and M. E. Hoff, Jr., 1960 IRE (Institute of Radio Engineers) WESCON (Western Electric Show and Convention) Convention Record, Pt. 4, pp. 96–104.
[42] L. Lönnblad, C. Peterson, and T. Rögnvaldsson, Comput. Phys. Commun. **70**, 167 (1992).
[43] A. Bohr and B. R. Mottelson, *Nuclear Structure* (W. A. Benjamin, New York, 1969), Vol. I.
[44] B. W. Mathews, Biochim. Biophys. Acta **405**, 442 (1975).
[45] H. Bohr and S. Brunak, *Protein Structure by Distance Analysis* (IOS Press, Amsterdam, 1994).
[46] H. G. Bohr, *Neural Network Prediction of Protein Structures* (Polyteknisk Forlag, Lyngby, Denmark, 1998).