Journal of Mathematical Chemistry 6(1991)63–76

AN APPLICATION OF NEURAL NETWORKS IN CHEMISTRY. PREDICTION OF ^{13}C NMR CHEMICAL SHIFTS

V. KVASNIČKA
Department of Mathematics, Slovak Technical University, 81237 Bratislava, Czechoslovakia

Received 12 April, 1990
(Revised 18 July 1990)

Abstract

Basic definitions of neural networks are given in terms of oriented graphs. Partial derivatives of an objective function with respect to the weight and threshold coefficients are derived. These derivatives are very important for the adaptation process, carried out by a version of the gradient method of the neural network considered. The stability of the adapted neural network toward small changes – "perturbation" – of input activities is described by sensitivities. The theory is illustrated by application of simple neural networks that reflect the topology of molecules to the classification of ^{13}C NMR chemical shifts of secondary carbons in acyclic alkanes.

1. Introduction

Neural networks [1–3] are computer-modeled or algorithmic systems derived from a simplified concept of the brain. In a neural network, a number of nodes, called neurons, are interconnected into a net-like structure. A network is constructed with

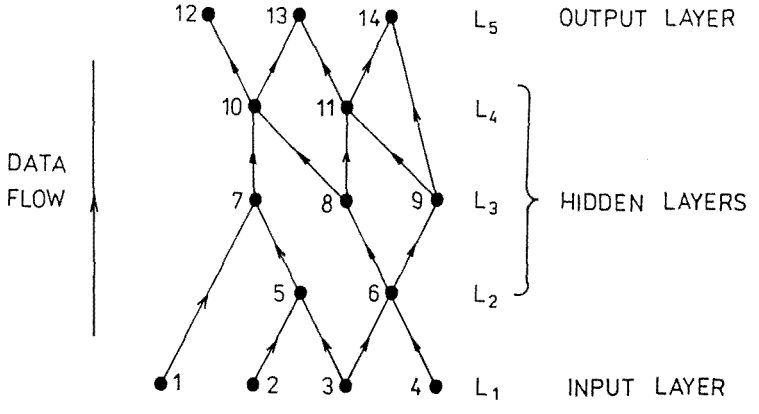

Fig. 1. Plot of a 5-layer neural network represented by an acyclic oriented graph composed of 14 vertices and 17 oriented edges.

three or more layers of neurons: input neurons, output neurons, and often one or more layers of intermediate elements called the hidden neurons (see fig. 1).

Each neuron receives input signals via one-way connections from preceding neurons, and each input is weighted by a variable weight parameter. If the sum of weighted inputs to a neuron exceeds a certain threshold coefficient, the neuron will send a signal to another neuron located in the juxtaposed higher layer.

Unlike conventional computers, neural networks are parallel in their structure and in the way they process information. The structure of the network, in particular the number of layers and the distribution of neurons among layers, is adjusted so that it is appropriate for the problem under study. The network is then put through a training (adaptation) process in which the weight and threshold coefficients are modified recursively by a learning algorithm, based on a "training set" of known data, until the weights and thresholds converge to fixed values. Assuming that the adaptation process was finished successfully, the formed network can be used to solve new problems in a so-called active process. Unlike the standard computer (Von Neumann's concept of the computer), knowledge is represented in a neural network in a parallel fashion in terms of weight and threshold coefficients distributed throughout the system.

The advantages of neural networks over conventional algorithms include (1) their *self-learning feature*, and (2) the *ability to generalize*. However, there are also a number of disadvantages. In particular, neural networks (1) are *poor at mathematics*, (2) sometimes *fail to give the correct answer*, and (3) *cannot explain their predictions*.

Neural networks do give incorrect answers, especially when they have been "trained" in a configuration with inappropriate weight and threshold coefficients from which they cannot escape or, more frequently, when they have been set up incorrectly. There may be a wrong assignment between the number of neurons in the network and the number of identifiable patterns in the data set. Another possible reason for failure may be an insufficient number of trials in the training process.

Neural networks are not a new "technology" [4]. The crucial point in this field is the concept of perceptron [5], widely used for many learning and pattern recognition algorithms. This concept was the subject of serious criticism [6], directed especially against its inability to solve certain more complex types of problems. The field of neural networks has seen a revival [7] in the 1980's, after it was realised that this criticism was only relevant to a very simple type of neural networks – perceptrons.

Applications of neural networks to chemistry have just begun to emerge [8]. The early ones [9,10] touch the problem of prediction of three-dimensional protein structure from data on amino acids. Chemical and Engineering News [8] reported a short review communication on a few initial applications of neural networks in organic chemistry, in particular, the distribution of products of nitration in a series of monosubstituted benzenes and the prediction of adverse drug effects. Recently, we have published [11] a preliminary communication on the application of a standard 3-layer neural network applicable to the classification of ^{13}C NMR chemical shifts of acyclic alkanes.

The purpose of the present communication is to suggest and elaborate a graph-theoretical formulation of neural networks that are appropriate for acyclic molecular

graphs, explicitly accounting for their topology. The theory will be illustrated by the evaluation of ^{13}C NMR chemical shifts of secondary carbon atoms in acyclic hydrocarbons.

2. Neural networks

In general, a neural network is determined as an oriented graph [12]

$$G = (V, E), \tag{1}$$

where $V = \{v_1, v_2, \ldots, v_N\}$ is a nonempty set composed of N vertices – *neurons* $v_1, v_2, \ldots, v_N$. The set $E = \{e_1, e_2, \ldots, e_M\}$ is composed of M edges – *connections* $e_1, e_2, \ldots, e_M$. Each connection $e \in E$ is formally interpreted (with respect to the set V) as an ordered pair of two distinct neurons $v, v' \in V$, $e = [v, v']$. We say that the connection $e = [v, v']$ is outgoing from the neuron v and incoming to the neuron v'. We shall postulate that the graph G is *acyclic* and does not contain *multiedges*. The neurons of V are unambiguously classified as

(1) *input neurons*, incident only with outgoing connections,

(2) *output neurons*, incident only with incoming connections,
and

(3) *hidden neurons*, incident at least with one incoming connection and one outgoing connection.

This means that the set V may be divided into three disjoint subsets V_I, V_O, and V_H that are composed of input, output, and hidden neurons, respectively,

$$V = V_I \cup V_O \cup V_H. \tag{2}$$

In our forthcoming considerations, we shall always assume that the subset V_H is nonempty, i.e. the neural network has at least one hidden neuron.

Another alternative way [13] to determine the oriented graph G consists of the mapping Γ which assigns to each neuron $v \in V$ a subset $\Gamma(v) \subseteq V$. Let $v' \in \Gamma(v)$ be a neuron from the subset $\Gamma(v)$; then the ordered pair $[v, v']$ is an oriented connection of G. The subset $\Gamma(v)$ is composed of the so-called *successors* of neuron v in graph G. An "inverse" mapping Γ^{-1} assigns to each $v \in V$ a subset $\Gamma^{-1}(v) \subseteq V$ composed of the so-called *predecessors* of neuron v in graph G. In other words, if $v' \in \Gamma^{-1}(v)$, then the ordered pair $[v', v]$ is an oriented connection of G. Using these two mappings Γ and Γ^{-1}, the above classification of neurons and the corresponding decomposition (2) of the set V may be done alternatively as

$$V_I = \{v \in V, \Gamma(v) \neq \emptyset \quad \text{and} \quad \Gamma^{-1}(v) = \emptyset\}, \tag{3a}$$

$$V_O = \{v \in V;\ \Gamma(v) = \varnothing \quad \text{and} \quad \Gamma^{-1}(v) \neq \varnothing\}, \tag{3b}$$

$$V_H = \{v \in V;\ \Gamma(v) \neq \varnothing \quad \text{and} \quad \Gamma^{-1}(v) \neq \varnothing\}, \tag{3c}$$

where the symbol $\varnothing$ denotes the empty subset.

The decomposition (2) of V into three disjoint subsets corresponding to input, output, and hidden neurons may be more deeply specified in terms of the so-called *layers*. First, we have to introduce the notion of *distance* $d(v)$ between a neuron $v \in V_H$ and the input neurons from the subset V_I. This distance is determined as the length of the *longest* oriented path connecting the neuron $v \in V_H$ with an input neuron from V_I. Then, the subset V_H is divided into $(p-2)$ disjoint subsets called the *layers*,

$$V_H = L_2 \cup L_3 \cup \ldots \cup L_{p-1}, \tag{4a}$$

$$L_i = \{v \in V_H;\ d(v) = i - 1\}, \tag{4b}$$

where the positive integer $p - 2$ corresponds to the maximal distance between hidden neurons and input neurons. Setting $L_p = V_O$ (output layer) and $L_1 = V_I$ (input layer), we obtain the following decomposition of V in the layers:

$$V = L_1 \cup L_2 \cup \ldots \cup L_{p-1} \cup L_p. \tag{5}$$

If two neurons $v, v' \in V$ are linked by an oriented connection $[v, v'] \in E$, then $v \in L_i$, $v' \in L_{i'}$, where $1 \leq i < i' \leq p$; i.e. neuron v' should belong to a higher layer than neuron v. In other words, the neural network may be understood as a hierarchically (horizontally) structured oriented graph, in which going from the bottom to the top one first strikes the input layer L_1, then successively the hidden layers $L_2, L_3, \ldots, L_{p-1}$, and finally, at the top, the output layer L_p.

Finally, to establish the definition of the neural network in terms of the oriented (acyclic) graph G, we introduce two evaluations of its connections and neurons that are carried out by mappings φ and ψ. The mapping

$$\varphi : E \to \mathbb{R} \tag{6a}$$

evaluates each connection of G by a real number ($\mathbb{R}$ is the set of real numbers) called the *weight coefficient*. Let $e = [v_i, v_j] \in E$ be a connection of G; then the weight coefficient assigned to this connection is denoted by w_{ji},

$$\varphi([v_i, v_j]) = w_{ji}. \tag{6b}$$

The mapping

$$\psi : V_H \cup V_O \to \mathbb{R} \tag{7a}$$

assigns to each hidden or output neuron a real number called the *threshold coefficient*,

$$\psi(v_i) = \vartheta_i, \tag{7b}$$

where $v_i \in V_H \cup V_O$.

In summary, the *neural network* is graph-theoretically determined by an oriented acyclic graph (without multiedges) with connections evaluated by weight coefficients, hidden and output neurons evaluated by threshold coefficients. For simplicity and to avoid some formal difficulties, we shall assume that the neurons are indexed in such a way that input neurons come first, followed by hidden neurons, and finally output neurons. This convention means that the set V may be formally put equal to a set $\{1, 2, \ldots, N\}$, where N is the number of neurons in G. Let N_I, N_H, and N_O be the numbers of input, hidden, and output neurons, respectively ($N = N_I + N_H + N_O$); then

$$V_I = \{1, 2, \ldots, N_I\}, \tag{8a}$$

$$V_H = \{N_I + 1, N_I + 2, \ldots, N_I + N_H\}, \tag{8b}$$

$$V_O = \{N_I + N_H + 1, N_I + N_H + 2, \ldots, N\}. \tag{8c}$$

The neurons of G are additionally evaluated by the so-called *activities*, i.e. we assign to each $v_i \in V$ its activity, denoted by x_i. We shall postulate that the activities of input neurons remain constant, while for hidden and output neurons they are calculated recurrently by

$$x_i = f\left(\sum_{j \in \Gamma^{-1}(i)} w_{ij} x_j + \vartheta_i \right), \tag{9}$$

for each $i \in L_k$, where the index k is successively increased from 2 to p (i.e. $k = 2, 3, \ldots, p$). The summation runs over all indices of the subset $\Gamma^{-1}(i)$ assigned to the neuron indexed by i. These activities constitute the so-called *state vector* $x = (x_1, x_2, \ldots, x_N)$. The *transfer function* $f(\xi)$ is a positive and monotonically increasing function which fulfills asymptotic conditions $f(\xi) \to 1$ as $\xi \to \infty$ and $f(\xi) \to 0$ as $\xi \to -\infty$. For instance, these requirements are simply met if the transfer function $f(\xi)$ is given as

$$f(\xi) = \frac{1}{1 + \exp(-\xi)}, \tag{10}$$

its first derivative being determined by $f'(\xi) = f(\xi)[1 - f(\xi)]$. As was already mentioned, activities of input neurons are kept fixed in the course of recurrent calculation of all other activities. Going successively from layer L_2 to higher layers $L_3, L_4, \ldots, L_p$, we calculate first the activities of the hidden neurons and then, finally,

the activities of the output neurons. For fixed weight and threshold coefficients and prescribed activities of input neurons, such a successive calculation of the activities assigned to hidden and output neurons is called the *active process* of the neural network. In a similar fashion as in (2), the state vector x may be formally divided into three subvectors that are composed of input, hidden, and output activities,

$$x = x_{\mathrm{I}} \oplus x_{\mathrm{H}} \oplus x_{\mathrm{O}}, \tag{11}$$

where

$$x_{\mathrm{I}} = (x_1, x_2, \ldots, x_{N_{\mathrm{I}}}), \tag{12a}$$

$$x_{\mathrm{H}} = (x_{N_{\mathrm{I}}+1}, x_{N_{\mathrm{I}}+2}, \ldots, x_{N_{\mathrm{I}}+N_{\mathrm{H}}}), \tag{12b}$$

$$x_{\mathrm{O}} = (x_{N_{\mathrm{I}}+N_{\mathrm{H}}+1}, x_{N_{\mathrm{I}}+N_{\mathrm{H}}+2}, \ldots, x_N). \tag{12c}$$

In general, the neural network with fixed weight and threshold coefficients may be formally considered as a mapping

$$F : \mathbb{R}^{N_{\mathrm{I}}} \to (0,1)^{N_{\mathrm{O}}}, \tag{13a}$$

which assigns to an input state vector x_{I} an output state vector x_{O},

$$x_{\mathrm{O}} = F(x_{\mathrm{I}}). \tag{13b}$$

The hidden activities should not be explicitly displayed in this formula; they play only the role of intermediate results that are temporarily emerging in the recurrent calculation of the output activities. An explicit analytical form of the mapping F is easily constructed by successive use of (9) and (10).

Let us now concentrate on the so-called *adaptation (learning) process* of a neural network. For a prescribed pair $x_{\mathrm{I}}/\tilde{x}_{\mathrm{O}}$ of input and output state vectors, we try to find such weight and threshold coefficients that the actual neural-network response x_{O} on the prescribed input state vector x_{I} would be "closely related" to the prescribed output state vector $\tilde{x}_{\mathrm{O}}$. There exist several possible ways [3] to achieve the above-mentioned vague "close relationship". One of them is to minimize the *objective function*

$$E = \tfrac{1}{2}(x_{\mathrm{O}} - \tilde{x}_{\mathrm{O}})^2 = \tfrac{1}{2} \sum_{k \in L_p} (x_k - \tilde{x}_k)^2, \tag{14}$$

where $\tilde{x}_k$'s are entries of $\tilde{x}_{\mathrm{O}}$. This means that the goal of our adaptation process is to find such weight and threshold coefficients so as to minimize the objective function E. The so-called *back-propagation strategy* [3] of the adaptation process involves successive calculation of partial derivatives $\partial E/\partial w_{ij}$ and $\partial E/\partial \vartheta_j$ in going from the top

output layer to the bottom input layer. In order to calculate these partial derivatives, we have to know, first of all, the partial derivatives $\partial x_k/\partial w_{ij}$ and $\partial x_k/\partial \vartheta_j$, for each $k \in L_p \equiv V_O$. After simple but tedious manipulations, we obtain

$$\frac{\partial x_k}{\partial \vartheta_j} = \delta_{jk} x_k (1 - x_k), \tag{15a}$$

$$\frac{\partial x_k}{\partial \vartheta_j} = \left(\sum_{l \in \Gamma(j)} \frac{\partial x_k}{\partial w_{lj}} w_{lj} \right) (1 - x_j), \tag{15b}$$

$$\frac{\partial x_k}{\partial w_{ji}} = \frac{\partial x_k}{\partial \vartheta_j} x_i, \tag{15c}$$

where the symbol δ_{jk} corresponds to Kronecker's delta, $\delta_{jk} = 1$ for $j = k$ and $\delta_{jk} = 0$ for $j \neq k$. The first formula (15a) is satisfied for each $j \in V_O \equiv L_p$, whereas the second and third formulae are satisfied for each $j \in L_t$ and $t = p - 1, p - 2, \ldots, 2$, and the index i in (15c) fulfills $i \in \Gamma^{-1}(j)$. The partial derivatives of the objective function E determined by (14) are

$$\frac{\partial E}{\partial w_{ji}} = \sum_{k \in L_p} (x_k - \tilde{x}_k) \frac{\partial x_k}{\partial w_{ji}}, \tag{16a}$$

$$\frac{\partial E}{\partial \vartheta_j} = \sum_{k \in L_p} (x_k - \tilde{x}_k) \frac{\partial x_k}{\partial \vartheta_j}. \tag{16b}$$

Introducing (15a–c) into (16a–b), we obtain the final formulae for the first partial derivatives of the objective function E,

$$\frac{\partial E}{\partial \vartheta_j} = (x_j - \tilde{x}_j) x_j (1 - x_j), \tag{17a}$$

$$\frac{\partial E}{\partial \vartheta_j} = \left(\sum_{l \in \Gamma(j)} \frac{\partial E}{\partial w_{lj}} w_{lj} \right) (1 - x_j), \tag{17b}$$

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial \vartheta_j} x_i, \tag{17c}$$

with identical constraints on the indices i and j as in (15a–c). We see that, going step-by-step from the top to the bottom of the neural network, we can calculate successively the partial derivatives $\partial E/\partial w_{ji}$ and $\partial E/\partial \vartheta_j$ by applying these formulae.

The partial derivatives (17a–c) are useful in that they minimize the objective function E by making use of a version of the well-known gradient method [14]. In

our actual applications, the variable metric method [14,15] performs satisfactorily when the zero-step initial values of weight and threshold coefficients are randomly generated from the open interval $(-1, 1)$.

The above adaptation process of the neural network may be simply generalized also for more than one pair of prescribed input/output state vectors $\boldsymbol{x}_{\mathrm{I}}/\tilde{\boldsymbol{x}}_{\mathrm{O}}$, i.e. for q pairs of state vectors $\boldsymbol{x}_{\mathrm{I}}^{(1)}/\tilde{\boldsymbol{x}}_{\mathrm{O}}^{(1)}$, $\boldsymbol{x}_{\mathrm{I}}^{(2)}/\tilde{\boldsymbol{x}}_{\mathrm{O}}^{(2)}$, ..., $\boldsymbol{x}_{\mathrm{I}}^{(q)}/\tilde{\boldsymbol{x}}_{\mathrm{O}}^{(q)}$. Then the objective function is determined by

$$E = \sum_{i=1}^{q} E^{(i)}, \tag{18a}$$

$$E^{(i)} = \tfrac{1}{2}\left(\boldsymbol{x}_{\mathrm{O}}^{(i)} - \tilde{\boldsymbol{x}}_{\mathrm{O}}^{(i)}\right)^2, \tag{18b}$$

where $\boldsymbol{x}_{\mathrm{O}}^{(i)}$ is the output state vector of the neural network corresponding to the prescribed ith input state vector $\boldsymbol{x}_{\mathrm{I}}^{(i)}$ and $\tilde{\boldsymbol{x}}_{\mathrm{O}}^{(i)}$ the required (expected) response of the neural network assigned to $\boldsymbol{x}_{\mathrm{I}}^{(i)}$. The partial derivatives of E are then equal to the sum of the partial derivatives of $E^{(i)}$ evaluated by (17a–c).

In order to obtain a deeper insight into an adapted neural network (i.e. the weight and threshold coefficients are already chosen), we introduce the so-called *sensitivities* of the neural network. Let us assume that a given input state vector $\boldsymbol{x}_{\mathrm{I}}$ is changed as $\boldsymbol{x}_{\mathrm{I}} \to \boldsymbol{x}_{\mathrm{I}} + \Delta\boldsymbol{x}_{\mathrm{I}}$, where the "perturbation" $\Delta\boldsymbol{x}_{\mathrm{I}}$ is composed of entries that are of first-order smallness with respect to entries of $\boldsymbol{x}_{\mathrm{I}}$. Then the hidden and output entries of the state vector $\boldsymbol{x}$ are also changed as $\boldsymbol{x} \to \boldsymbol{x} + \Delta\boldsymbol{x}$. The hidden and output entries of $\Delta\boldsymbol{x}$ correspond to the response of the neural network to the "perturbation" $\Delta\boldsymbol{x}_{\mathrm{I}}$. When going successively from the second layer to higher layers, we may trace the propagation of the input "perturbation" $\Delta\boldsymbol{x}_{\mathrm{I}}$ throughout the whole neural network. The hidden and output entries of $\Delta\boldsymbol{x}$ are well approximated (up to first order) by

$$\Delta x_i \cong \sum_{j \in V_{\mathrm{I}}} \frac{\partial x_i}{\partial x_j} \Delta x_j, \tag{19}$$

for $i \in V_{\mathrm{H}} \cup V_{\mathrm{O}}$. The partial derivatives $S_{ij} = \partial x_i / \partial x_j$ are called the *sensitivities* of the neural network. These entities are simply calculable from (9–10),

$$S_{ij} = x_i(1 - x_i) \sum_{k \in \Gamma^{-1}(i)} w_{ik} S_{kj}, \tag{20}$$

for $i \in V_{\mathrm{H}} \cup V_{\mathrm{O}}$ and $j \in V_{\mathrm{I}}$. Entries $S_{kj} = \delta_{kj}$ if both indices $k, j \in V_{\mathrm{I}}$. The sensitivities, loosely speaking, are interpreted as follows: if entries S_{ij} have "relatively" very small numerical values, then the state vector $\boldsymbol{x}$ is "relatively" insensitive to the "perturbation" $\Delta\boldsymbol{x}_{\mathrm{I}}$; increasing values of S_{ij} indicate a greater sensitivity of the neural network to $\Delta\boldsymbol{x}_{\mathrm{I}}$, i.e. a small change in the input state vector $\boldsymbol{x}_{\mathrm{I}}$ may cause a considerable change in some hidden and output entries of the state vector $\boldsymbol{x}$.

3. Application

The theory of neural networks, outlined in the first part of this communication, will be illustrated by a special tree-like network used as a classifier of ^{13}C NMR chemical shifts of secondary carbon atoms in acyclic alkanes [16]. The physical phenomenon of chemical shifts represents a property well localized on atoms and influenced by other atoms from their environments mainly through the chemical bonds. This observation makes it possible to construct a neural network which will evaluate the ^{13}C NMR chemical shifts in a form closely related to the *topology* of the studied molecular systems. The form of the neural network used is displayed in fig. 2. It is composed of 27 neurons and 6 layers, and each neuron (except

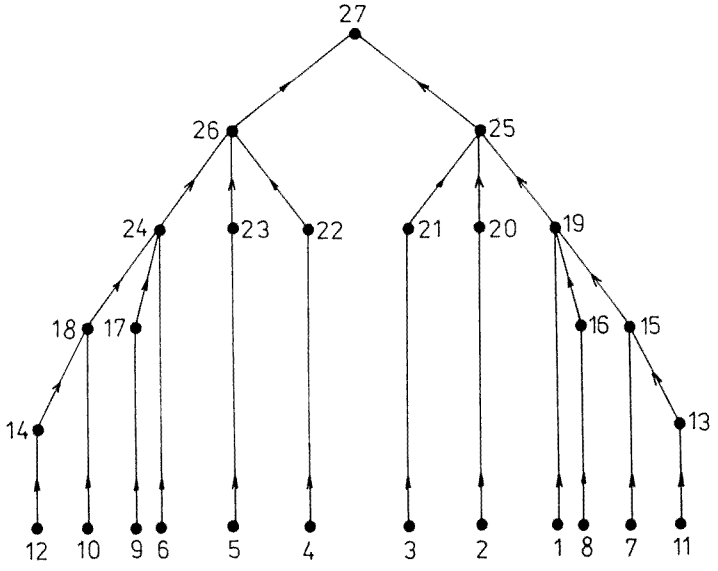

Fig. 2. Neural network used as a classifier of ^{13}C NMR chemical shifts of secondary carbons in acyclic alkanes. The activity of the output neuron (indexed by 27) is adapted such that it corresponds to the chemical shift of a classified secondary carbon atom. The dichotomic (0 or 1) input activities (i.e. descriptors) determine the topology of the classified molecule.

for the output neuron) has only one successor (i.e. for each $v \in V_I \cup V_H$ we have $|\Gamma(v)| = 1$). The dichotomic input activities x_i (for $i = 1, \ldots, 12$) specify the topology of classified molecules (cf. fig. 3 and table 1). If $x_i = 1$, then the edge $[i, \Gamma(i)]$ belongs to a subgraph of the neural network isomorphic to the classified molecule; in the opposite case (i.e. $x_i = 0$), this edge does not belong to

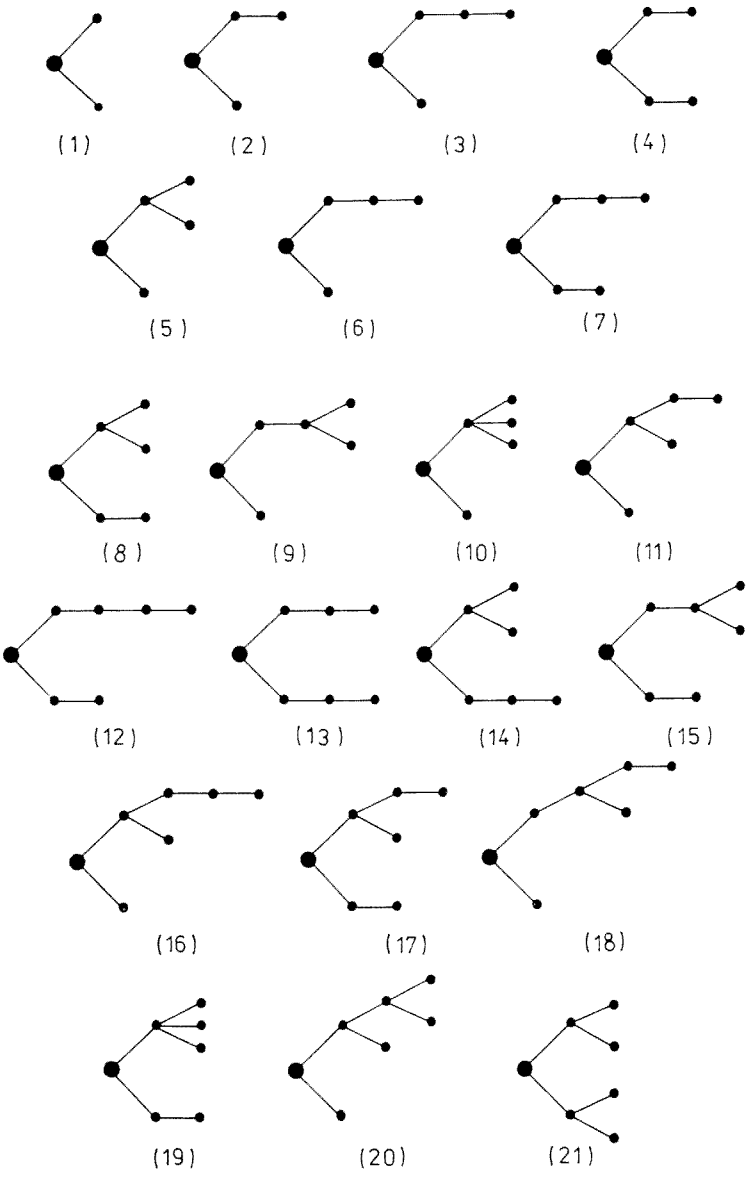

Fig. 3. Acyclic alkanes with secondary carbon atom (heavy dot). These molecules are graph-theoretically represented by rooted trees. The alkanes indexed 1 to 7 form the training set for the adaptation process of the neural network in fig. 1. The remaining alkanes (indexed 8 to 21) are used in the active process of the adapted neural network.

the subgraph. Examples of input activities of an alkane are given in fig. 4 and table 2. The training set is composed of 40 objects – subgraphs which are assigned to the first 7 alkanes in fig. 3.

Table 1

^{13}C NMR chemical shifts of secondary carbons in acyclic alkanes[a]

No.	δ_{exp} [b]	δ_{inc} [c]	δ_{nn} [d]	No.	δ_{exp}	δ_{inc}	δ_{nn}
1	16.3	15.9	17.5	12	32.0	32.5	32.0
2	24.9	25.3	25.9	13	29.0	29.7	29.1
3	22.2	22.8	22.4	14	38.9	39.1	38.3
4	34.1	34.7	34.2	15	29.7	29.7	28.8
5	31.6	32.2	31.6	16	29.5	29.4	30.4
6	22.7	23.1	22.7	17	39.0	39.1	39.5
7	31.7	32.2	31.6	18	20.2	20.6	19.9
8	41.9	41.6	40.9	19	47.3	46.0	43.6
9	20.8	20.3	19.6	20	26.8	27.2	28.4
10	36.1	36.6	34.3	21	49.0	48.5	47.6
11	29.4	29.7	30.2				

[a]The experimental values of chemical shifts are taken from ref. [16].

[b]The dashed line separates the chemical shifts that correspond to an object (secondary carbon atoms) from the training set.

[c]Chemical shifts calculated by an empirical incremental scheme [17].

[d]Chemical shifts produced by the neural network.

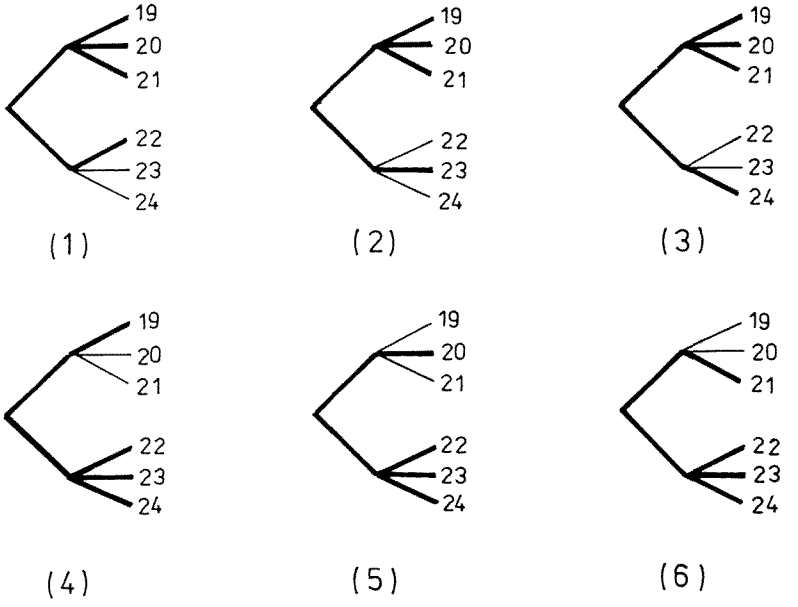

Fig. 4. Illustrative example of coding of acyclic alkanes with secondary carbons by the neural network in fig. 2. In general, what we are looking for in the neural network are all possible subgraphs of the classified alkane such that the secondary carbon (heavy dot) is superimposed on the output neuron. For instance, the alkane placed at the top of the figure has six distinct subgraphs in the neural network, where the network is, for simplicity, represented by its top 3 layers.

Table 2

Examples of input activities (descriptors) of the alkane indexed by 19 in fig. 3.

No.	1	2	3	4	5	6	7	8	9	10	11	12
1	1	1	1	1	0	0	0	0	0	0	0	0
2	1	1	1	0	1	0	0	0	0	0	0	0
3	1	1	1	0	0	1	0	0	0	0	0	0
4	1	0	0	1	1	1	0	0	0	0	0	0
5	0	1	0	1	1	1	0	0	0	0	0	0
6	0	0	1	1	1	1	0	0	0	0	0	0

The output activity, corresponding to the chemical shift, is a real entry from the open interval (0, 1). Here, we have to specify more precisely the actual meaning of the concept of output activity assigned to chemical shifts of carbon atoms. As follows from (10), the transfer function f maps the set $\mathbb{R}$ of real numbers onto the open interval (0, 1), i.e. $f: \mathbb{R} \rightarrow (0, 1)$. This means that the entries of the output state vector x_O belong merely to the interval (0, 1). That is, the required output vector $\tilde{x}_O$ should also be composed merely of real numbers from the open interval (0, 1). Since the chemical shifts of carbon atoms are, say, ranged within $5 \leq \delta \leq 50$, this interval should be compressed by an analog of (10) to a subinterval of (0, 1). Such a mapping was, in our studies, realized by

$$y = g(x) = \frac{1}{1 + \exp(-ax - b)}, \tag{21a}$$

where constants a and b are determined as follows:

$$a = \frac{A_{min} - A_{max}}{x_{min} - x_{max}}, \quad b = -ax_{min} - A_{min}, \tag{21b}$$

$$A_{min} = \ln \frac{y_{min}}{1 - y_{min}}, \quad A_{max} = \ln \frac{y_{max}}{1 - y_{max}}. \tag{21c}$$

The entries x_{min} and x_{max} correspond to minimal and maximal values, respectively, of the chemical shifts (in our case, we set $x_{min} = 5$ and $x_{max} = 50$), whereas the entries y_{min} and y_{max} are minimal and maximal values of the "compressed" chemical shifts (we assign $y_{min} = 0.05$ and $y_{max} = 0.95$); for these actual values of minimal and maximal entries, the constants a and b are $a = 0.1308640$ and $b = -3.5987588$. An inverse transformation of (21a) is

$$x = g^{-1}(y) = \frac{1}{a}\left(\ln \frac{y}{1 - y} - b\right). \tag{22}$$

This function is used for the "back" transformation of output activities of the neural network to the actual chemical shifts.

The optimal values of weight and threshold coefficients resulting from the adaptation process are listed in table 3. The values of chemical shifts evaluated by the adapted neural network are given in table 1. Simple inspection of these results

Table 3

Optimal values of weight and threshold coefficients*

Weight coefficients			
$w_{1,19} = -1.603$	$w_{2,20} = 2.282$	$w_{3,21} = 2.282$	$w_{4,22} = 2.282$
$w_{5,23} = 2.282$	$w_{6,24} = 2.282$	$w_{7,15} = 1.254$	$w_{8,16} = -2.327$
$w_{9,17} = -2.327$	$w_{10,18} = 1.254$	$w_{11,13} = -0.919$	$w_{12,14} = 0.919$
$w_{13,15} = 0.954$	$w_{14,18} = 0.954$	$w_{15,19} = 1.964$	$w_{16,19} = -1.047$
$w_{17,24} = -1.047$	$w_{18,24} = 1.964$	$w_{19,25} = -3.869$	$w_{20,25} = 2.886$
$w_{21,25} = 2.886$	$w_{22,26} = 2.886$	$w_{23,26} = 2.886$	$w_{24,26} = 3.869$
$w_{25,27} = 3.474$	$w_{26,27} = 3.474$		
Threshold coefficients			
$\vartheta_{13} = 0.004$	$\vartheta_{14} = 0.004$	$\vartheta_{15} = 0.077$	$\vartheta_{16} = 0.054$
$\vartheta_{17} = 0.054$	$\vartheta_{18} = 0.077$	$\vartheta_{19} = 0.075$	$\vartheta_{20} = -0.901$
$\vartheta_{21} = -0.901$	$\vartheta_{22} = -0.901$	$\vartheta_{23} = -0.901$	$\vartheta_{24} = 0.075$
$\vartheta_{25} = 0.037$	$\vartheta_{26} = 0.037$	$\vartheta_{27} = -3.487$	

* Value of objective function $E = 1.842 \times 10^{-4}$.

allows one to conclude that the used neural network provides chemical shifts that are closely related to their experimental values. For completeness, the chemical shifts calculated empirically by an incremental scheme [17] are also given in table 1.

4. Discussion

We have demonstrated that the approach of neural networks offers a simple novel tool potentially well suited for the classification of molecular properties. The present form of neural networks – an oriented acyclic graph – provides a very effective computational device for the classification of physical or physico-chemical properties that are well localized on single atoms and influenced by their environments through the chemical bonds. The suggested theory of neural networks is based on the assumption that the "information flow" is carried out through those connections that form a subgraph isomorphic to the classified molecular object. In the present communication, we have illustrated the theory on acyclic alkanes classified with respect to the ^{13}C NMR chemical shifts of secondary carbon atoms. The results

obtained by other authors [8–10] and also the results presented here indicate that neural networks can give valuable predictions. Neural network approaches do not replace other forms of computing predictions, but they promise to be a useful alternative tool for computationally approaching problems that would not be satisfactorily solved by standard numerical methods.

The present theory of neural networks may be simply and straightforwardly generalized for molecules involving also multiple bonds and heteroatoms. Our preliminary results and experience [18] indicate that this enlarged neural-network approach offers, for classes of structurally similar molecules, a very effective computing tool for the classification of molecular systems with respect to a preselected physical property. The neural networks program, written in PASCAL for IBM PC/AT compatibles, is available on request.

References

[1] *IEEE 1st Conf. on Neural Networks,* vols. 1–4, San Diego, CA (1987),
[2] R. Eckmiller and C.v.D. Malsburg (eds.), *Neural Computers* (Springer, Berlin, 1988).
[3] D.E. Rumelhart and J.L. McClelland, *Parallel Distributed Processes,* vols. 1–2 (MIT Press, Cambridge, MA, 1986).
[4] W.S. McCulloch and W. Pitts, Bull. Math. Biophys. 5(1943)115.
[5] F. Rosenblatt, *Principles of Neurodynamics* (Spartan Books, Washington, DC, 1962).
[6] M.L. Minsky and S.A. Papert, *Perceptrons* (MIT Press, Cambridge, MA, 1969).
[7] J.J. Hopfield, Proc. Natl. Acad. Sci. USA 79(1982)2554.
[8] S. Borman, Chem. Eng. News (April, 1989) 24.
[9] Ningh Qian and T.J. Sejnowski, J. Mol. Biol. 202(1988)865.
[10] L.H. Holley and M. Karplus, Proc. Natl. Acad. Sci. USA 86(1989)152.
[11] V. Kvasnička, Chem. Papers, in press.
[12] F. Harary, *Graph Theory* (Addison–Wesley, Reading, MA, 1969).
[13] C. Berge, *Théorie des Graphes et ses Applications* (Gauthier–Villars, Paris, 1958).
[14] E. Polak, *Computational Methods in Optimization* (Academic Press, New York, 1971).
[15] R. Fletcher and M.J.D. Powell, Comput. J. 6(1963)163.
[16] H.-O. Kalinowski, S. Berger and S. Braun, *^{13}C NMR Spektroskopie* (G. Thieme, Stuttgart, 1984).
[17] D.M. Grant and E.G. Paul, J. Amer. Chem. Soc. 86(1964)2984.
[18] V. Kvasnička and J. Pospíchal, J. Mol. Struct. (THEOCHEM), submitted.