Applied Intelligence 12, 117–146 (2000)

Application of Cascade Correlation Networks for Structures to Chemistry

ANNA MARIA BIANUCCI
Dipartimento di Scienze Farmaceutiche, Via Bonanno 6, 56126, Pisa, Italy

ALESSIO MICHELI, ALESSANDRO SPERDUTI AND ANTONINA STARITA
Dipartimento di Informatica, Università di Pisa, Corso Italia, 40, 56125, Pisa, Italy

Abstract. We present the application of Cascade Correlation for structures to QSPR (quantitative structure-property relationships) and QSAR (quantitative structure-activity relationships) analysis. Cascade Correlation for structures is a neural network model recently proposed for the processing of structured data. This allows the direct treatment of chemical compounds as labeled trees, which constitutes a novel approach to QSPR/QSAR. We report the results obtained for QSPR on Alkanes (predicting the boiling point) and QSAR of a class of Benzodiazepines. Our approach compares favorably versus the traditional QSAR treatment based on equations and it is competitive with 'ad hoc' MLPs for the QSPR problem.

Keywords: Cascade Correlation networks, constructive algorithms, gradient descent, QSPR, QSAR

1. Introduction

In several application domains, the information is organized in structured representations. These representations are particularly suited to capture the nature of the relationships between basic entities of the problem at hand. Examples of application domains where structures are extensively used are medical and technical diagnoses (discovery and manipulation of structured dependencies, constraints, explanations), molecular biology (DNA and protein analysis), chemistry (classification of chemical structures, quantitative structure-property relationship (QSPR), quantitative structure-activity relationship (QSAR)), automated reasoning (robust matching, manipulation of logical terms, proof plans, search space reduction), software engineering (quality testing, modularization of software), geometrical and spatial reasoning (robotics, structured representation of objects in space, figure animation, layouting of objects), speech and text processing (robust parsing, semantic disambiguation, organizing and finding structure in texts and speech).

While algorithms that manipulate symbolic information are capable of dealing with highly structured data, they very often are not able to deal with noise and incomplete data. Moreover, they are usually not suited to deal with domains where both categorical (symbols) and numerical entities coexist and have the same relevance for the solution of the problem.

Neural networks are universally recognized as tools suited for dealing with noise and incomplete data, especially in contexts where numerical variables play a relevant role in the solution of the problem. In addition to this capability, when used for classification and/or prediction tasks, they do not need a formal specification of the problem, just requiring a set of examples showing samples of the function to be learned. Unfortunately, neural networks are mostly regarded as learning models for domains in which instances are organized into *static* data structures, like records or fixed-size arrays, and thus they do not seem suited to deal with structured domains. Recurrent neural networks, that generalize feedforward networks to sequences (a particular case of dynamically structured data) are perhaps the best known exception.

In recent years, however, there has been some effort in trying to extend the computational capabilities of neural networks to structured domains. Different

approaches have been proposed. Touretzky's BoltzCONS system [1] is an example of how a Boltzman machine can handle symbolic structures using coarse-coded memories [2] as basic representational elements, and LISP's *car*, *cdr*, and *cons* functions as basic operations. The RAAM model proposed by Pollack [3] is based on backpropagation[1] to discover compact recursive distributed representations of trees with a fixed branching factor. Recursive distributed representations are an instance of the concept of a reduced descriptor introduced by Hinton [4] to solve the problem of mapping part-whole hierarchies into connectionist networks. Also related to the concept of reduced descriptor are Plate's holographic reduced representations [5]. A formal characterization of representations of structures in connectionist systems using the tensor product was developed by Smolensky [6].

While these earlier approaches were able to deal with some aspects of processing of structured information, none of them established a practical and efficient way of dealing with structured information. A more powerful approach, at least for classification and prediction tasks, was proposed in [7] and further extended in [8]. In these works a generalization of recurrent neural networks for processing sequences to the case of directed graphs is presented. The basic idea behind this generalization is the extension of the concept of *unfolding* from the domain of sequences to the domain of directed ordered graphs (DOGs). We will give more details on these type of neural networks for the class of directed ordered acyclic graphs (DOAGs) in Section 3.

The possibility of processing structured information using neural networks is appealing for several reasons. First of all, neural networks are universal approximators; in addition, they are able to learn from a set of examples and very often, by using the correct methodology for training, they are able to reach a quite high generalization performance. Finally, as already mentioned above, they are able to deal with noise and incomplete, or even ambiguous, data.

All these capabilities are particularly useful when dealing with prediction tasks where data is usually gathered experimentally, and thus is partial, noisy, and incomplete. A typical example of such a domain is chemistry, where compounds can naturally be represented as labeled graphs. Each node of the graph is an atom or a group of atoms, while edges represent bonds between atoms. So chemistry seems to be the right domain where to test the computational capabilities of neural networks for processing of structures. In fact, one fundamental problem in chemistry is the prediction of both the physical properties and the biological activity of chemical compounds. In the former case, we speak of *Quantitative Structure-Property Relationship* (QSPR), while in the latter case we speak of *Quantitative Structure-Activity Relationship* (QSAR). Let us give a closer look at QSAR in order to understand why neural networks for processing of structures should be chosen as computational tools to perform the prediction. The biological activity of a drug is fully determined by the micromechanism of interaction of the active molecules with the bioreceptor. Unfortunately, discovering this micromechanism is very hard and expensive. Hence, because of the assumption that there is a direct correlation between the activity and the structure of the compound, the QSAR approach is a way of approaching the problem by comparing the structure of all known active compounds with inactive compounds, focusing on similarities and differences between them. The aim is to discover which substructure or which set of substructures characterize the biomechanism of activity, so as to generalize this knowledge to new compounds. This new knowledge would enable us to design new drugs on the basis of the known structure-activity relationships supplied by the QSAR analysis, allowing a more effective use of the resources. The earliest attempts to find relationships between molecular properties of biologically active compounds and their activities were performed since the past century. A systematic approach to the treatment of these relationships was mainly introduced by Hansch et al. in the 60s [9–11] with the development of equations able to correlate the biological activity to physical and chemical properties of biologically active compounds. Several different models were then developed based on equations exploiting a wide variety of molecular properties, including structural descriptors such as topological indices [12]. QSPR can be considered as a generalization of the QSAR concept. In fact it assumes that general properties, such as physical properties, of the compounds can be related to their chemical and morphological structure. Feed-forward neural networks have been applied with different modalities to QSPR and QSAR [13], as explained in Section 2.2. However, just as traditional QSPR/QSAR approaches, the standard approach with feedforward neural networks consists of encoding each graph as a fixed-size vector of features, which is then used for feeding the neural network. The features used may involve physico-chemical properties [14], or topological indices, or vectorial graph codes [15]. Unfortunately, the a priori definition of the

encoding process has several drawbacks. For example, when the encoding is performed by using *topological indices*, they need to be properly designed by an *expert* through a very expensive *trial and error approach*. Thus this approach needs an expert, which may not be available, or may be very expensive, or even may be misleading if the expert knowledge is not correct, and in addition a very expensive procedure must be followed in order to select the topological indices which are relevant for the computational task. Changing the class of chemical compounds under study, or the computational task, will of course mean that all the above steps must be performed from scratch. Moreover, the resulting vectorial representations of graphs obtained in this way may be very difficult to classify.

Neural networks for structures [7] face this problem by simultaneously learning both how to represent and how to classify structured patterns. This is a very desirable property which has already been shown to be useful in practice [16]. For this reason, the aim of the present work is to demonstrate that by using neural networks for structures it is possible to directly correlate the biological activity or physical property to the molecular structure of the compound. In fact, neural networks for structures are capable to directly process molecular structures represented as labeled directed ordered acyclic graphs. The specificity of the proposed approach stems from the ability of these networks to automatically encode the structural information depending on the computational problem at hand, i.e., the representation of the molecular structures is not defined a priori, but learned on the basis of the training set. This ability is proved in this paper by the application of Cascade Correlation for structures [7, 17] to two radically different QSAR/QSPR problems: the prediction of the non-specific activity (affinity) towards the Benzodiazepine/$GABA_A$ receptor by a group of Benzodiazepines (Bz) [11], and the prediction of the boiling point for a group of acyclic hydrocarbons (Alkanes) [15].

The paper is organized as follows. Different approaches to QSPR/QSAR are reviewed in Section 2. Specifically, topological indices are presented in Section 2.1, while the application of standard neural networks to chemistry is discussed in Section 2.2. Neural networks for the processing of structures, called *recursive neural networks*, are briefly described in Section 3, where details on the Cascade Correlation model for structures, used in this paper, are given as well. The computational tasks faced in this paper, i.e., the prediction of the affinity towards the Benzodiazepine/$GABA_A$ receptor, and the prediction of the boiling point for Alkanes, are explained in Section 4. Representational issues for chemical compounds are discussed in Section 5 and experimental results obtained by using these representations in combination with the Cascade Correlation model are reported in Section 6. Discussion and conclusions are contained in Section 7 and Section 8, respectively. At the end of the paper there is an appendix reporting the datasets used for the experiments, and detailed results obtained for each compound.

2. Approaches to QSPR/QSAR

The basic assumption of QSPR/QSAR is that there exists a direct relationship between the chemical structure of a compound and its physical properties or its biological activity with respect to a receptor. Moreover, it is assumed that this relationship can be quantified.

A typical way of representing the information about a chemical structure is to resort to chemical graphs, i.e., graphs where each node corresponds to an atom (or functional group of atoms) and each edge corresponds to a chemical bond. The main problem in traditional approaches to QSPR/QSAR analysis is to find a good numerical representation capable of retaining the chemical and topological information present in chemical graphs and to relate it to the target property, e.g., biological activity. The need for a numerical representation is due to the use of mathematical models, e.g., multi-linear regression, to quantify the relationship of specific structural features of a compound with the target value.

Different approaches to QSPR/QSAR can be distinguished according to the way these numerical representations are defined and/or devised. Basically, two different methods have been proposed for the definition of numerical descriptors:

- using known *physico-chemical properties* [18], such as polarizability, molar refractivity, hydrophobicity, etc.; these parameters basically measure physical and chemical properties of atoms or groups of atoms, such as substituents, giving a partial description of the molecular structure;
- using *topological indices*, which code specific morphological properties of the molecular graph.

These descriptors are then related to the target property, and in particular to biological activity, by more

traditional techniques, such as multiple linear regression, principal component analysis (PCA), partial least square (PLS) [19], or more recently developed techniques, such as genetic algorithms [20, 21] and neural networks [22].

In the following, we briefly review topological indices and neural networks techniques for QSPR/QSAR since they are directly related to the approach proposed in this work.

2.1. Topological Indices

Topological indices have been developed to represent structural features (graph invariants) on the basis of graph theory. To this end, molecular structure is expressed in numerical form suitable for manipulation in algebraic equations. In this approach, the focus of attention is on geometrical and topological features rather than chemical ones. For example, topological indices typically convey information about the "shape" and "size" of chemical compounds.

The topological index based methodology for QSPR/QSAR is typically defined through a 3 step procedure:

1. the chemical compound is represented by using a simplified structure derived from the molecular skeleton by taking off both hydrogen atoms and the typology of bond, atoms and groups;
2. the above representation is encoded by numerical indices that represent specific topological features;
3. one or more meaningful indices are selected, according to the problem at hand, and used as input to regression (or classification) models, such as linear regression.

Specifically, concerning step 2, the count of atoms, or bonds, or couple of adjacent bonds, are among the simplest descriptors. These descriptors were used since the very beginning to evaluate the qualitative relationship between structure and property. Unfortunately, they are not powerful enough to allow the discrimination of different molecular topologies. For example, they are not able to fully capture the branching structure of a given compound. The theory of topological indices tries to remedy this representational gap by defining more complex and powerful descriptors able to retain specific additional structural information depending on the property to be predicted. Due to the different possibilities of relationship between topological characteristics and properties, a considerable number of topological indices have been developed. Among these there are adjacency or distance matrix based indices [23–25] and indices based on information theory [26]. A systematic classification of topological indices based on the encoded morphological characteristics can be found in [12].

As an example, here we discuss the derivation of the *chi* connectivity indices family. The key concept in *chi* indices is the decomposition of the molecular graph into fragments (subgraphs) of different size and complexity. A *chi* index is calculated as the weighted count of a given type of subgraph. The order and type of the index depend on the size and complexity of the selected subgraph, respectively. The order m of the index is the number of graph edges in the considered subgraph, whereas the type t refers to the particular arrangement of the edges (e.g., Path (P), Cluster (C), Path/Cluster(PC), and Ring (R); see Fig. 1) in the subgraph.

In order to compute a *chi* index of a molecule, each vertex of the corresponding molecular structure needs to be represented by a numerical value, the so called *delta value*:

$$\delta_i = |S(i)| \tag{1}$$

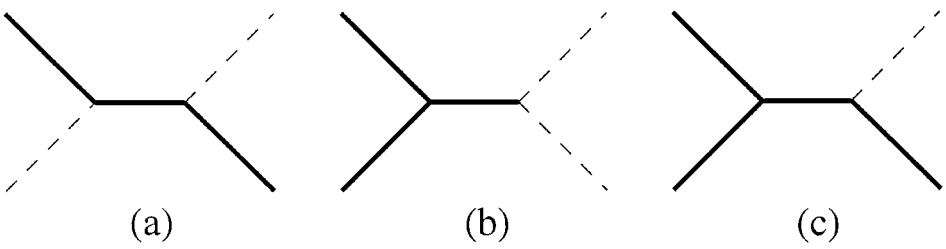


Figure 1. Different types of subgraphs within a structure composed by 6 nodes. The subgraphs are shown in bold: (a) path of order 3; (b) *cluster* of order 3; (c) path/*cluster* of order 4.

where $|S(i)|$ is the degree of the ith vertex. The *delta value* is then used to define the contribution of single subgraphs to the overall computation of the index.

For example, in the *chi* index of zero order ($^0\chi$) the selected subgraph is a single vertex. Zero degree denotes the absence of edges in fragments containing only single vertexes. By defining the contribution term c for a given subgraph s as

$$^0c_s = (\delta_s)^{-1/2}, \tag{2}$$

the index can be computed as the sum of these terms for all subgraphs (vertexes) in the graph:

$$^0\chi = \sum {}^0c_s. \tag{3}$$

In first order *chi* index ($^1\chi$), edges are taken in account as graph fragments. In analogy with Eqs. (2) and (3) c and χ are defined as:

$$^1c_s = (\delta_i \delta_j)_s^{-1/2} \tag{4}$$

$$^1\chi = \sum {}^1c_s \tag{5}$$

where an edge s connects vertex i with vertex j, and the sum is over the set of edges.

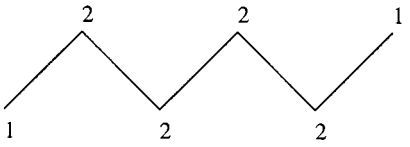
Eventually, indices of higher order ($m > 1$) exploit more complex types (t) of graph fragments, such as paths and clusters:

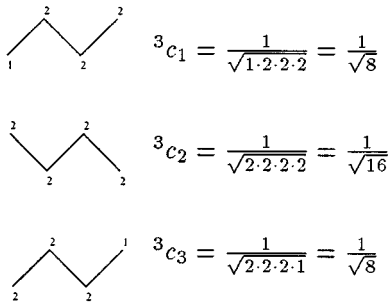
$$^mc_s = \left(\prod(\delta_i)_s\right)^{-1/2} \tag{6}$$

$$^m\chi_t = \sum {}^mc_s \tag{7}$$

where the product is over all delta values in each singular fragment of order m, and the sum is over all fragments of order m and type t.

Examples of computations of $^3\chi_p$ are shown in Fig. 2. Note that, for each chemical structure, several distinct topological indices can be computed. The choice of how many indices and which specific one is



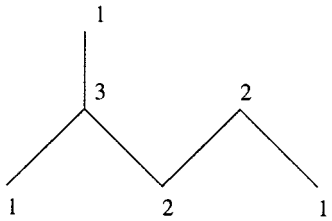


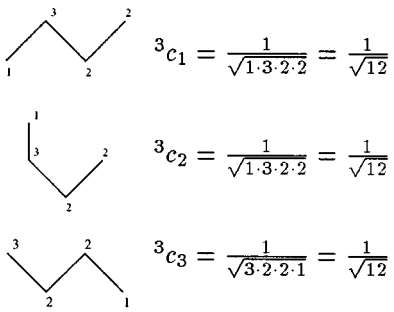
$$^3c_1 = \frac{1}{\sqrt{1\cdot2\cdot2\cdot2}} = \frac{1}{\sqrt{8}}$$

$$^3c_2 = \frac{1}{\sqrt{2\cdot2\cdot2\cdot2}} = \frac{1}{\sqrt{16}}$$

$$^3c_3 = \frac{1}{\sqrt{2\cdot2\cdot2\cdot1}} = \frac{1}{\sqrt{8}}$$

$$^3\chi_p = {}^3c_1 + {}^3c_2 + {}^3c_3 = \frac{1}{\sqrt{8}} + \frac{1}{\sqrt{16}} + \frac{1}{\sqrt{8}} = 0.957$$





$$^3c_1 = \frac{1}{\sqrt{1\cdot3\cdot2\cdot2}} = \frac{1}{\sqrt{12}}$$

$$^3c_2 = \frac{1}{\sqrt{1\cdot3\cdot2\cdot2}} = \frac{1}{\sqrt{12}}$$

$$^3c_3 = \frac{1}{\sqrt{3\cdot2\cdot2\cdot1}} = \frac{1}{\sqrt{12}}$$

$$^3\chi_p = {}^3c_1 + {}^3c_2 + {}^3c_3 = \frac{1}{\sqrt{12}} + \frac{1}{\sqrt{12}} + \frac{1}{\sqrt{12}} = 0.866$$

Figure 2. Examples of computations for $^3\chi_p$ topological indices. There are three possible paths composed of three edges in each represented structure. For every vertex the delta values are reported and we calculate the ^3c_s value for each fragment. It can be noted that different structures lead to different indices values.

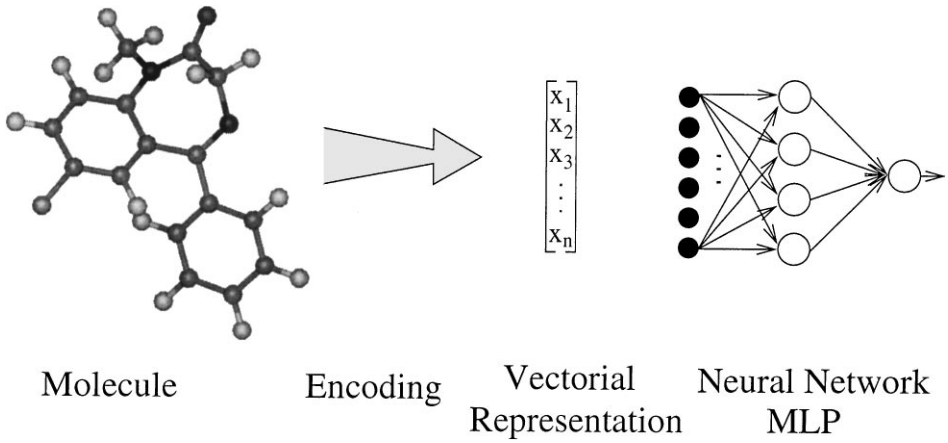


Figure 3. Typical setting for neural networks application to QSPR/QSAR.

computed depends both on the type of chemical compounds and on the computational task.

For different classes of molecules, the use of topological indices in linear regression has been successful, showing that they are able to retain topological information which is directly correlated with the target property. In effect, the usefulness of the topological approaches for certain groups of problems, mostly for Alkanes and Hydrocarbons, is beyond doubt. However, they cannot be recognized as general solution methods for many other QSAR problems. For example, it remains under discussion both how to expand the definition of the indices for families of compounds containing heteroatoms and bounds of different order, and a unique numerical characterization for each molecule in large data sets.

Examples of properties studied in this context are solubility, boiling point, density, molar volume, molecular refraction, partition coefficients and thermodynamic properties. Some QSAR applications have been developed for toxicity analysis, anesthetic potency, hallucinogenic activity, enzyme inhibition, flavor, odor and taste, antimicrobial and antibacterial activity, carcinogenicity. Some examples of the above applications are reported in [12].

2.2. Neural Networks

Recently, feed-forward neural networks have been used in QSPR and QSAR, since they constitute a more powerful regression tool than multiple linear regression. In fact, the nonlinearity of neural networks can be exploited to better correlate topological indices and physico-chemical properties to target properties and activities [14, 15, 27, 28].

The usual setting for the application of neural networks to QSPR and QSAR is shown in Fig. 3. The molecule is typically represented (output of the encoding phase) by a vector of features which may be both topological indices and physico-chemical properties. As we will see in the following, a bit more sophisticated representations have also been proposed. The vectorial representation is then fed in input to a Multilayer Perceptron (MLP) for classification or prediction. Different neural networks approaches basically differ in the way the molecule is represented.

The most typical approach is the one where a feedforward neural network is trained on physico-chemical properties or numerical topological indices (see [13, 23] for reviews). This approach has been used for tasks where a high degree of nonlinearity is required for the development of the QSPR/QSAR analysis, as in the case of QSAR of a class of Benzodiazepines [27, 29]. An example of a QSPR study which fits in this context has been the prediction of the boiling point and other properties of Alkene [30], where besides using χ indices, other 'ad hoc' topological indices are defined in order to deal with double bonds which occur in the unsaturated hydrocarbons family and to distinguish isomers of Alkenes.

Some authors tried to preserve more information on the molecular structure through a *vectorial* or *matricial* representation of the chemical graph. An approach of this type is used in [15], were the authors present a study on the prediction of the boiling points of Alkanes. In this case, the input to the MLP is given by a vectorial

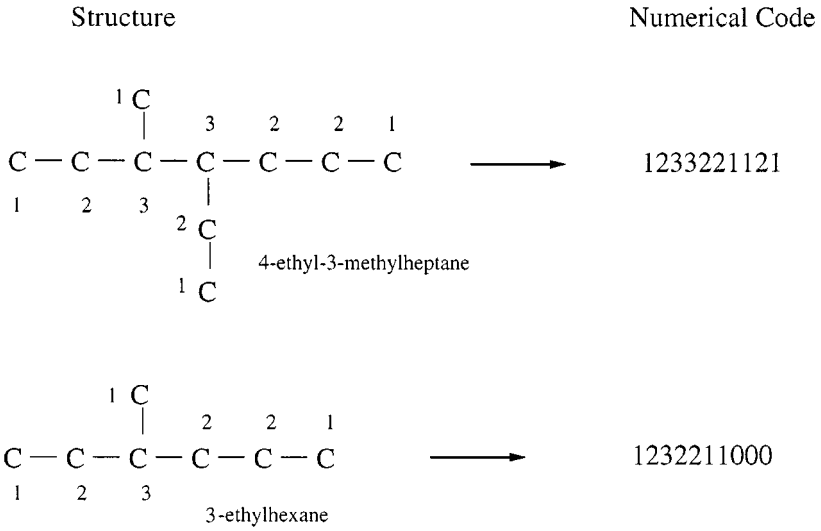


Figure 4. Example of derivation of the vectorial code for two Alkanes. The vectorial code is obtained starting from a chemical graph where hydrogen atoms are "suppressed". The numbers represent the degree of each node in the graph.

code obtained by encoding the chemical graph with suppressed hydrogens through a "*N-tuple*" code (see Fig. 4). Each component of the vectorial code, which in this case is of dimension 10 since only Alkanes with up to 10 atoms are considered, represents the number of carbon bonds for each atom. The uniqueness of the code is guaranteed by keeping a lexicographic order.

This representation for Alkanes is particularly efficient for the prediction of the boiling point since it is well known that the boiling point is strongly correlated with the number of carbon atoms and the branching of the molecular structure. Of course, while the obtained predictions are very good, the same representation could be useless for a different class of compounds. An example of matricial representation can be found in [31], where Elrod et al. used simplified and fixed size connectivity matrices (5×5) to partially represent the structure of a substituent.

It must be noted that both vectorial and matricial representations have a fixed dimension and thus they cannot fully encode structures of larger size. Moreover, smaller structures need to be represented by inserting zeros into the void positions. The first problem could be solved by enlarging the size of the representation, however this would increase the number of free parameters of the MLP, thus decreasing the probability to obtain a good generalization capability, since the data sets are typically very small in size and usually it is not easy to extend them to compensate for the increased number of parameters. The latter problem leads to the underutilization of parameters.

Another problem of vectorial and matricial representations, however, must be recognized as the uniqueness of representation. In fact, in order to avoid that the same compound is represented in different ways, a procedure for the assignment of a unique enumeration for the vertexes of the chemical graph must be devised. In [31], the problem is partially solved by numbering the atoms[2] in a substituent according to a breadth first visit of the corresponding chemical graph starting from the point where the substituent is attached (see Fig. 5). Any additional ambiguity is ignored. Whereas this approach seems to work for the family of compounds examined in this study, there is no guarantee to obtain a unique representation for a larger set of compounds. In order to obtain a unique representation for each compound, it would be necessary to devise a canonical enumeration procedure. Unfortunately, such

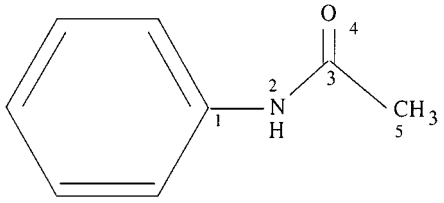


Figure 5. Example of numbering assigned to each atom of the C_aminoacetyl moiety of the acetanilide.

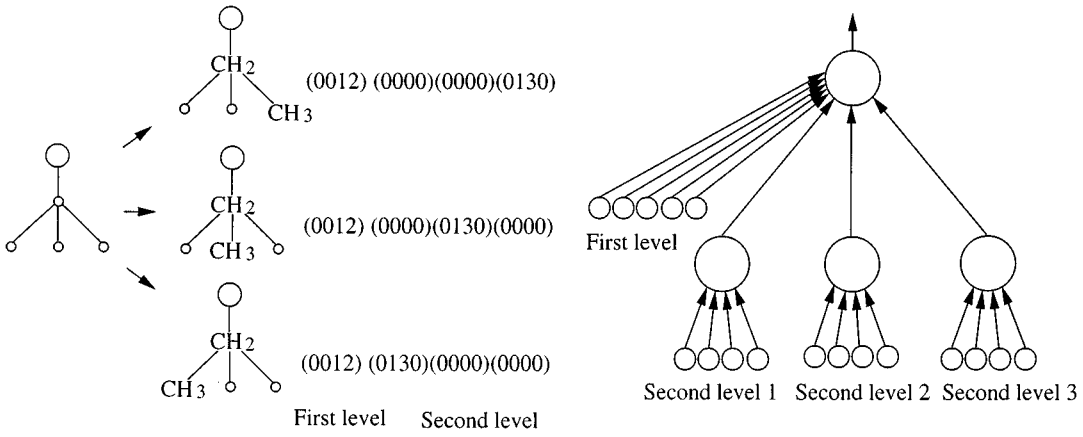


Figure 6. The topology of the neural network architecture (on the right side) reproduces the template (rooted trees with two levels) superimposed on a portion of a singular substituent. The value of the external inputs are the result of a vectorial coding process, showed on the left side for the functional group-CH_2CH_3, using physico-chemical information. The same compound may be superimposed in three different ways onto the input units.

procedure should solve the more general problem of isomorphism between graphs, which is known to be computable in polynomial time only for some classes of graphs.

Finally, a template based approach has been explored as well [32]. The basic idea of this approach is to have a neural network which mimics the chemical structure of the input compound. A common or given *template* in the family of compounds to be examined is individuated a priori and then used as topology for the neural network. Specifically, each input unit in the network corresponds to an atom (or to a functional group of atoms) into a specific position of the molecular template and it allows to signal the presence (or absence) of a chemical entity in that position of the template. Moreover, the network is not fully connected, since connections are present in correspondence with bonds represented into the template (see Fig. 6 for an example). Of course, because of the common template, this approach can be applied only for a set of very homogeneous compounds.

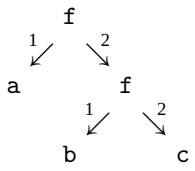
3. Neural Networks for Structures

In this section we present a general framework for the processing of structures by neural networks. First of all, we introduce some preliminary notions on graphs which are also needed to fix the notation and the working conditions. Then we give the main idea underpinning the approach and present the computation and learning models for a Cascade Correlation based network, which is used for QSPR and QSAR analysis in the following.

3.1. Preliminaries on Graphs

Here we consider structured domains which are sets of labeled directed ordered acyclic graphs (DOAGs). For a DOAG we mean a DAG $\boldsymbol{Y}$ with vertex set vert($\boldsymbol{Y}$) and edge set edg($\boldsymbol{Y}$), where for each vertex $v \in$ vert($\boldsymbol{Y}$) a total order on the edges leaving from v is defined. Labels are tuples of variables and are attached to vertices. The void DOAG will be denoted by the special symbol ξ.

For example, in the case of graphs representing logical terms, the order on outgoing edges is immediately induced by the order of the arguments to a function; e.g., the logical term `f(a,f(b,c))` can be represented as:



We shall require the DOAG either to be empty or to possess a supersource, i.e., a vertex $s \in$ vert($\boldsymbol{Y}$) such that every vertex in vert($\boldsymbol{Y}$) can be reached by a directed path starting from s. Note that if a DOAG does not possess a supersource, it is still possible to define a convention for adding an extra vertex s (with a minimal number of outgoing edges), such that s is a supersource

for the expanded DOAG [7]. The function *source*($\boldsymbol{Y}$) returns the (unique) supersource of $\boldsymbol{Y}$.

The *outdegree* of a node v is the cardinality of the set of outgoing edges from v, while the *indegree* of v is the cardinality of the set of edges incident on v. In the following, a generic class of DOAGs with labels in $\mathcal{I}$ and bounded (but unspecified) indegree and outdegree, will be denoted by $\mathcal{I}^{\#}$.

3.2. Recursive Neural Networks

Recursive neural networks [7] are neural networks able to perform mappings from a set of labeled graphs to the set of real vectors. Specifically, the class of functions which can be realized by a recursive neural network can be characterized as the class of functional graph transductions $\mathcal{T}: \mathcal{I}^{\#} \to \mathbb{R}^k$, where $\mathcal{I} = \mathbb{R}^n$, which can be represented in the following form

$$\mathcal{T} = g \circ \hat{\tau}, \tag{8}$$

where $\hat{\tau}: \mathcal{I}^{\#} \to \mathbb{R}^m$ is the *encoding* (or *state transition*) function and $g: \mathbb{R}^m \to \mathbb{R}^k$ is the *output* function. Specifically, given a DOAG $\boldsymbol{Y}$, $\hat{\tau}$ is defined recursively as

$$\hat{\tau}(\boldsymbol{Y}) = \begin{cases} \mathbf{0} \text{ (the null vector in } \mathbb{R}^m) & \text{if } \boldsymbol{Y} = \xi \\ \tau\left(s, \boldsymbol{Y}_s, \hat{\tau}\left(\boldsymbol{Y}^{(1)}\right), \ldots, \hat{\tau}\left(\boldsymbol{Y}^{(o)}\right)\right) & \text{otherwise} \end{cases} \tag{9}$$

where the *c-model* function τ is defined as

$$\tau : V \times \mathbb{R}^n \times \underbrace{\mathbb{R}^m \times \cdots \times \mathbb{R}^m}_{o \text{ times}} \to \mathbb{R}^m \tag{10}$$

where V is the set of all vertices, $\mathbb{R}^n$ denotes the label space, while the remaining domains represent the encoded subgraphs spaces up to the maximum outdegree of the input domain $\mathcal{I}^{\#}$, o is the maximum outdegree of DOAGs in $\mathcal{I}^{\#}$, $s = \textit{source}(\boldsymbol{Y})$, $\boldsymbol{Y}_s$ is the label attached to the supersource of $\boldsymbol{Y}$, and $\boldsymbol{Y}^{(1)}, \ldots, \boldsymbol{Y}^{(o)}$ are the subgraphs pointed by s. The function τ is called c-model function since it defines a computational model for the encoding function.

Note that, because of Eq. (9), $\mathcal{T}$ is *causal* since τ only depends on the current node and nodes descending from it. Moreover, when τ does not depend on any specific vertex, i.e., $\tau(\boldsymbol{Y}_s, \hat{\tau}(\boldsymbol{Y}^{(1)}), \ldots, \hat{\tau}(\boldsymbol{Y}^{(o)}))$, then $\mathcal{T}$ is also *stationary*. In this paper we focus on stationary transductions.

Example 1 (Encoding of logical terms). Given a stationary encoding function $\hat{\tau}$, the encoding of the logical term `f(a,f(b,c))` is defined by the following set of equations

$$\hat{\tau}\left(\begin{array}{ccccc} & & \texttt{f} & & \\ & \overset{1}{\swarrow} & & \overset{2}{\searrow} & \\ \texttt{a} & & & & \texttt{f} \\ & & & \overset{1}{\swarrow} & \overset{2}{\searrow} \\ & & \texttt{b} & & \texttt{c} \end{array}\right) = \tau\left(\texttt{f}, \hat{\tau}(\texttt{a}), \hat{\tau}\left(\begin{array}{ccc} & \texttt{f} & \\ \overset{1}{\swarrow} & & \overset{2}{\searrow} \\ \texttt{b} & & \texttt{c} \end{array}\right)\right),$$

$$\hat{\tau}\left(\begin{array}{ccc} & \texttt{f} & \\ \overset{1}{\swarrow} & & \overset{2}{\searrow} \\ \texttt{b} & & \texttt{c} \end{array}\right) = \tau(\texttt{f}, \hat{\tau}(\texttt{b}), \hat{\tau}(\texttt{c})),$$

$$\hat{\tau}(\texttt{b}) = \tau(\texttt{b}, \textit{nil}, \textit{nil}),$$
$$\hat{\tau}(\texttt{a}) = \tau(\texttt{a}, \textit{nil}, \textit{nil}),$$
$$\hat{\tau}(\texttt{c}) = \tau(\texttt{c}, \textit{nil}, \textit{nil}),$$

where a, b, and c denote the graphs with a single node labeled `a`, `b`, and `c`, respectively.

Concerning the output function g, it can be defined as a map

$$g: \mathbb{R}^m \to \mathbb{R}^k. \tag{11}$$

Note that Eqs. (10) and (11) only describe the general form for τ and g. Different realizations can be given which satisfy the above equations. For example, both τ and g can be implemented by feedforward neural networks. Before to reach such level of complexity, however, it is worth to explore simpler realizations. Specifically, let us study what happens for a single recursive neuron with $m = 1$. The simplest non-linear neural realization for $\tau(\cdot)$ is given by

$$\begin{aligned} x &= \tau\left(\boldsymbol{l}, x^{(1)}, \ldots, x^{(o)}\right) \\ &= f\left(\sum_{i=1}^{n} w_i l_i + \sum_{j=1}^{o} \hat{w}_j x^{(j)} + \theta\right), \end{aligned} \tag{12}$$

where f is a sigmoidal function, w_i are the weights associated to the label space, $\hat{w}_j$ are the weights associated to the subgraphs spaces, θ is the bias, $\boldsymbol{l}$ is the current input label, $x^{(1)}, \ldots, x^{(o)}$ are the encoded representations of subgraphs (recall that $x^{(j)} = \hat{\tau}(\boldsymbol{Y}^{(j)})$, and

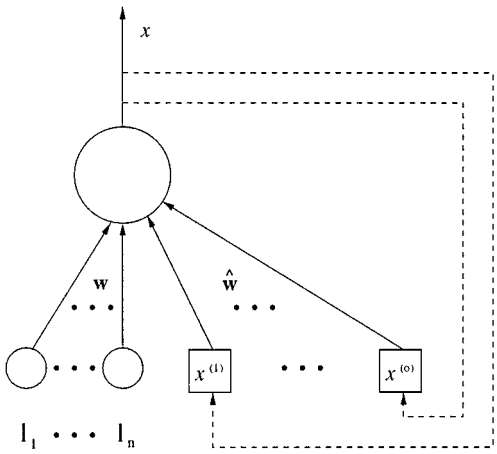


Figure 7. A graphical representation of a recursive neuron.

x is the encoding of the current structure). A graphical representation of the single recursive neuron is given in Fig. 7.

To clarify the use of recursive neurons, let us consider the tree `f(a,f(b,c))` in Example 1. The encoding function $\hat{\tau}$(`f(a,f(b,c))`) can be implemented by unfolding the recursive neuron (i.e., Eq. (12)) on the structure `f(a,f(b,c))`, giving rise to a feed-forward network called *encoding network* (see Fig. 8). Note that the weights of the recursive neuron are repeated according to the topology of the structure. The neural representation of the structure, i.e., $\hat{\tau}$(`f(a,f(b,c))`), is given by the output of the *encoding network*.

When $m > 1$, $\tau(\cdot) \in \mathbb{R}^m$ can be written as

$$\tau\left(\boldsymbol{l}, \boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(o)}\right) = \boldsymbol{F}\left(\boldsymbol{W}\boldsymbol{l} + \sum_{j=1}^{o} \hat{\boldsymbol{W}}_j \boldsymbol{x}^{(j)} + \boldsymbol{\theta}\right), \tag{13}$$

where $\boldsymbol{F}_i(v) = f(v_i)$ (sigmoidal function), $\boldsymbol{l} \in \mathbb{R}^n$ is the label attached to the current vertex, $\boldsymbol{\theta} \in \mathbb{R}^m$ is the bias vector, $\boldsymbol{W} \in \mathbb{R}^{m \times n}$ is the weight matrix associated with the label space, $\boldsymbol{x}^{(j)} \in \mathbb{R}^m$ are the vectorial codes obtained by the application of the encoding function $\hat{\tau}$ to the subgraphs $\boldsymbol{Y}^{(j)}$, and $\hat{\boldsymbol{W}}_j \in \mathbb{R}^{m \times m}$ is the weight matrix associated with the jth subgraph space.

Concerning the output function $g(\cdot)$, it can be defined as a set of standard neurons taking as input the encoded representation $\boldsymbol{x}$ of the graph, i.e.,

$$g(\boldsymbol{x}) = \boldsymbol{F}(\boldsymbol{M}\boldsymbol{x} + \boldsymbol{\beta}), \tag{14}$$

where $\boldsymbol{M} \in \mathbb{R}^{k \times m}$ and $\boldsymbol{\beta} \in \mathbb{R}^k$ are the weight matrix and bias terms defining $g(\cdot)$, respectively.

3.3. *Learning with Cascade Correlation for Structures*

In this section we discuss how a neural graph transduction $\mathcal{T}$ can be learned using an extension of the Cascade Correlation algorithm. The standard Cascade Correlation algorithm [33] creates a neural network using an incremental approach for the classification (or

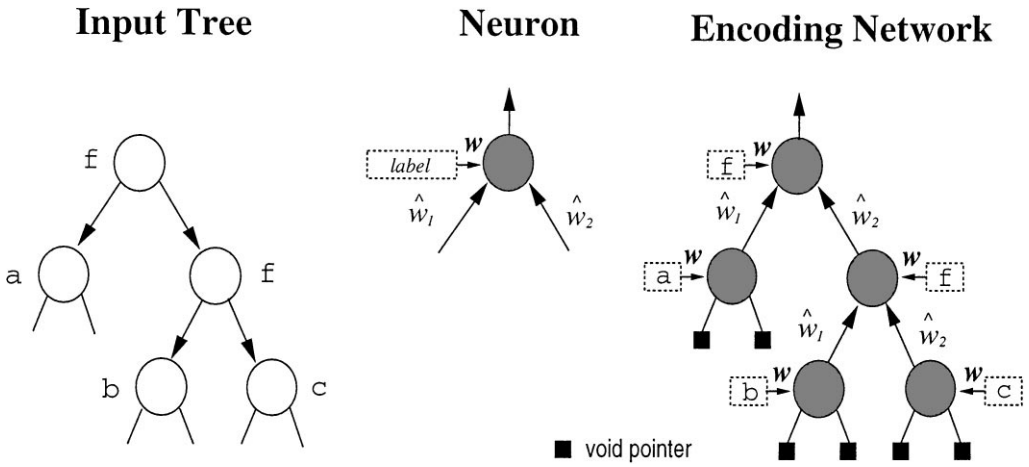


Figure 8. Given the input structure shown in the left side of the figure, and a recursive neuron with $m = 2$ and $o = 2$, the encoding network shown in the right side is generated. The black squares represent void pointers which are encoded as null vectors (in this case, the void pointer is equal to 0). Note that the weights of the encoding network are copies of the weights of the recursive neuron. The labels, here represented as symbols, are supposed to be encoded through suitable numerical vectors.

regression) of unstructured patterns. The starting network $\mathcal{N}_0$ is a network without hidden nodes trained by a Least Mean Square algorithm. If network $\mathcal{N}_0$ is not able to solve the problem, a hidden unit u_1 is added such that the *correlation* between the output of the unit and the residual error of network $\mathcal{N}_0$ is maximized.[3] The weights of u_1 are frozen and the remaining weights are retrained. If the obtained network $\mathcal{N}_1$ cannot solve the problem, new hidden units are added which are connected (with frozen weights) with all the inputs and previously installed hidden units. The resulting network is a *cascade* of nodes. Fahlman extended the algorithm to the classification of sequences, obtaining good results [34].

In the following, we show that the Cascade Correlation can be further extended to structures by using our computational scheme [17]. In fact, the shape of the c-model function can be expressed component-wise by the following set of equations:

$$\begin{aligned}
\tau_1 &= h_1\big(\boldsymbol{l}, \hat{\tau}_1\big(\boldsymbol{Y}^{(1)}\big), \ldots, \hat{\tau}_1\big(\boldsymbol{Y}^{(o)}\big)\big),\\
\tau_2 &= h_2\big(\boldsymbol{l}, \hat{\tau}_1\big(\boldsymbol{Y}^{(1)}\big), \ldots, \hat{\tau}_1\big(\boldsymbol{Y}^{(o)}\big),\\
&\qquad \hat{\tau}_2\big(\boldsymbol{Y}^{(1)}\big), \ldots, \hat{\tau}_2\big(\boldsymbol{Y}^{(o)}\big), \hat{\tau}_1(\boldsymbol{Y})\big),\\
&\;\;\vdots\\
\tau_m &= h_m\big(\boldsymbol{l}, \hat{\tau}_1\big(\boldsymbol{Y}^{(1)}\big), \ldots, \hat{\tau}_1\big(\boldsymbol{Y}^{(o)}\big),\\
&\qquad \hat{\tau}_2\big(\boldsymbol{Y}^{(1)}\big), \ldots, \hat{\tau}_2\big(\boldsymbol{Y}^{(o)}\big), \ldots,\\
&\qquad \hat{\tau}_m\big(\boldsymbol{Y}^{(1)}\big), \ldots, \hat{\tau}_m\big(\boldsymbol{Y}^{(o)}\big),\\
&\qquad \hat{\tau}_1(\boldsymbol{Y}), \ldots, \hat{\tau}_{m-1}(\boldsymbol{Y})\big), \qquad (15)
\end{aligned}$$

where the h_i are suitable nonlinear functions of the arguments.

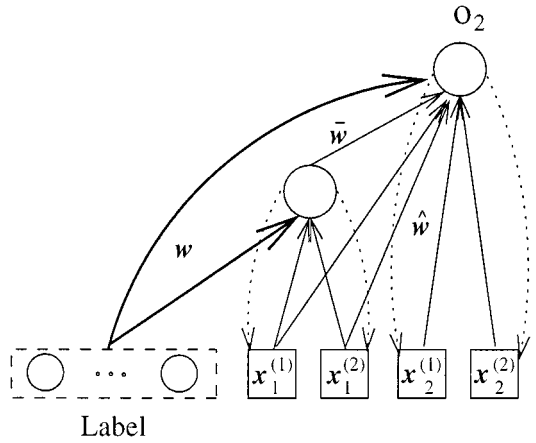


Figure 9. Architectural elements in a Cascade Correlation for Structure (CCS) network with $m = 2$ and $o = 2$.

An example of a network implementing the above equations for the case $m = 2$ and $o = 2$ is illustrated in Fig. 9. The evolution of a network, obtained by adding a new hidden recursive neuron at each main iteration of the algorithm, is shown in Fig. 10.

Specifically, the output of the kth hidden unit, in our framework, can be computed as

$$\begin{aligned}
o_k &= \tau_k\big(\boldsymbol{l}, \boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(o)}\big)\\
&= f\left(\sum_{i=0}^{n} w_i^{(k)} l_i + \sum_{v=1}^{k}\sum_{j=1}^{o} \hat{w}_{(v,j)}^{(k)} \boldsymbol{x}_v^{(j)}\right.\\
&\qquad \left. + \sum_{q=1}^{k-1} \bar{w}_q^{(k)} \tau_q\big(\boldsymbol{l}, \boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(o)}\big)\right), \qquad (16)
\end{aligned}$$

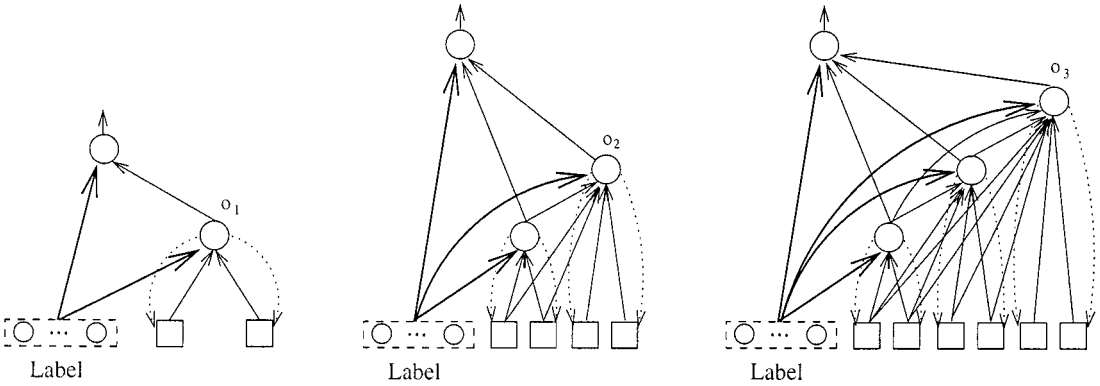


Figure 10. The evolution of a network with $o = 2$. The output of the network is meaningful only when the activity of the hidden units represents the code for a complete structure.

where $w_0^{(k)} = \theta$ and $l_0 = 1$, $w_{(v,j)}^{(k)}$ is the weight (of the kth hidden unit) associated with the output of the vth hidden unit computed on the jth subgraph code $\boldsymbol{x}^{(j)}$, and $\bar{w}_q^{(k)}$ is the weight of the connection from the qth (frozen) hidden unit, $q < k$, and the kth hidden unit. The output of the network (with k inserted hidden units) is then computed according to Eq. (14), where $\boldsymbol{M} \in \mathbb{R}^k$ since we have a single output unit. Moreover, since we are interested in biological activity prediction, the output unit is set to be linear, i.e., $g(\boldsymbol{x}) = \boldsymbol{M}^t \boldsymbol{x} + \beta$.

Learning is performed as in standard Cascade Correlation by interleaving the minimization of the total error function (LMS) and the maximization of the correlation of the new inserted hidden unit with the residual error. The main difference with respect to standard Cascade Correlation is in the calculation of the derivatives. According to Eq. (16), the derivatives of $\tau_k(\boldsymbol{l}, \boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(o)})$ with respect to the weights are computed as

$$\frac{\partial \tau_k\left(\boldsymbol{l}, \boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(o)}\right)}{\partial w_i^{(k)}} = f'\left(l_i + \sum_{j=1}^{o} \hat{w}_{(k,j)}^{(k)} \frac{\partial \boldsymbol{x}_k^{(j)}}{\partial w_i^{(k)}}\right)$$

$$\frac{\partial \tau_k\left(\boldsymbol{l}, \boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(o)}\right)}{\partial \bar{w}_q^{(k)}} = f'\left(\tau_q\left(\boldsymbol{l}, \boldsymbol{x}^{(1)}, \ldots, \mathrm{x}^{(o)}\right) + \sum_{j=1}^{o} \hat{w}_{(k,j)}^{(k)} \frac{\partial \boldsymbol{x}_k^{(j)}}{\partial \bar{w}_q^{(k)}}\right)$$

$$\frac{\partial \tau_k\left(\boldsymbol{l}, \boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(o)}\right)}{\partial \hat{w}_{(v,t)}^{(k)}} = f'\left(\boldsymbol{x}_v^{(t)} + \sum_{j=1}^{o} \hat{w}_{(k,j)}^{(k)} \frac{\partial \boldsymbol{x}_k^{(j)}}{\partial \hat{w}_{(v,t)}^{(k)}}\right)$$

where $i = 0, \ldots, n$, $q = 1, \ldots, (k-1)$, $v = 1, \ldots, k$, $t = 1, \ldots, o$, f' is the derivative of $f(\ )$. The above equations are recurrent on the structures and can be computed by observing that for graphs composed by a single vertex $\frac{\partial \boldsymbol{x}_k}{\partial w_i^{(k)}} = l_i f'$, $\frac{\partial \boldsymbol{x}_k}{\partial \bar{w}_q^{(k)}} = \boldsymbol{x}_q f'$ for $q < k$, and all the remaining derivatives are null. Consequently, we only need to store the output values of the unit and its derivatives for each component of a structure.

Learning for the output weights proceeds as in the standard Cascade Correlation algorithm.

4. QSPR/QSAR Tasks

In order to evaluate the prediction capability of neural networks for structures in QSPR/QSAR applications, we have selected one QSAR task, i.e., the prediction of the non-specific activity (affinity) towards the Benzodiazepine/GABA$_A$ receptor by a group of Benzodiazepines (Bz) (classical 1,4-benzodiazepin-2-ones) [11], and one QSPR task, i.e., the prediction of the boiling point for a group of acyclic hydrocarbons (Alkanes) [15]. These tasks have been selected in order to have a direct comparison of our approach with both equational approaches and 'ad hoc' feedforward neural networks, respectively.

4.1. QSAR Task: Benzodiazepines

The ability of predicting the biological activity of chemical compounds showing therapeutic interest constitutes the major aspect of the drug design. Benzodiazepines, for example, has been extensively studied since the 70s, as this class of compounds plays the major role in the field of minor tranquilizers, and several QSAR studies have been carried out aiming at the prediction of the non-specific activity (affinity) towards the Benzodiazepine/GABA$_A$ receptor. The biological activity can be expressed as the logarithm of the inverse of the drug concentration C (Mol./liter) able to give a fixed biological response.[4]

As a first approach, a group of Benzodiazepines (Bz) (classical 1,4-benzodiazepin-2-ones) previously analyzed by Hadjipavlou-Litina and Hansch [11] through the traditional QSAR equations, was analyzed.[5] The data set analyzed by Hadjipavlou-Litina and Hansch (see Table 2 of [11]) appeared to be characterized by a good molecular diversity, and this last requirement makes it particularly significant in any kind of QSAR analysis. For this reason, we have used the same data set. The total number of molecules was 77.

The analyzed molecules present a common structural aspect given by the Benzodiazepine ring and they differ each other because of a large variety of substituents at the positions showed in Fig. 11.

4.2. QSPR Task: Alkanes

To assess the true performance of standard neural networks in QSPR, they are usually tested on well known physical properties. A typical example is given by the prediction of the boiling point of Alkanes. The prediction task is well characterized for this class of compounds, since the boiling points of hydrocarbons depend upon molecular size and molecular shape, and vary regularly within a series of compounds, which means that there is a clear correlation between molecular shape and boiling point. Moreover, the relatively simple structure of these compounds[6] is amenable to

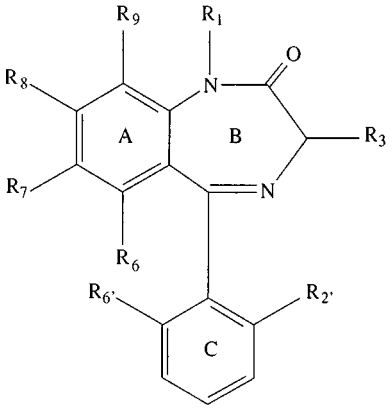


Figure 11. The common template shared by the majority of the analyzed molecules.

very compact representations such as topological indices and/or vectorial codes, which are capable of retaining the relevant information for prediction. For these reasons, multilayer feed-forward networks using 'ad hoc' representations yield very good performances.

In order to perform a comparison with our method, we decided to use as reference point the work described in [15] which uses multilayer feed-forward networks. The data set used in [15] comprised all the 150 Alkanes with up to 10 carbon atoms. It must be noted that Cherqaoui et al. use a vectorial code representation of Alkanes based on the n-tuple code for the encoding of trees (see Fig. 4). So they represent each Alkane as a 10 numerical components vector with the last components filled by zeros when the number of atoms of the compound is less than 10. The single component encodes the number of bonds of the corresponding carbon node.

5. Representation of the Molecular Structure

The main requirement for the use of the Cascade Correlation for structures network consists in finding a representation of molecular structures in terms of DOAGs. The candidate representation should retain the detailed information about the topology of the compound, atom types, bond multiplicity, chemical functionalities, and finally it should show a good similarity with the representations usually adopted in chemistry. The main representational problems are: to represent cycles, to give a direction to edges, to define a total order over the edges. An appropriate description of the molecular structures analyzed in this work is based on a labeled tree representation. In fact, concerning the first problem, since cycles are absent in the examined Alkanes and they mainly constitute the common shared template of the benzodiazepines compounds, it is reasonable to represent a cycle (or a set of connected cycles, as in the benzodiazepines case) as a single node where the attached label carries information about its chemical nature. The second problem was solved by the definition of a set of rules based on the I.U.P.A.C. nomenclature system.[7] Finally, the total order over the edges follows a set of rules mainly based on the size of the sub-compounds.

5.1. Representation of Benzodiazepines

The labeled tree representation of a Benzodiazepine is obtained by the following minimal set of rules:

1. the root of the tree represents the Bz ring;
2. the root does have as many subtrees as substituents on the Bz ring, sorted according to the order conventionally followed in pharmaceutical chemistry;
3. each atom (or cycle) of a substituent is represented by a node, and each bond[8] by an edge; the root of the subtree representing the substituent corresponds to the atom directly connected to the common template, and the orientation of the edges follows the increasing levels of the trees;
4. suitable labels, representing the atom type (or cycle), are associated to the root and to all the nodes;
5. the total order on the subtrees of each node is hierarchically defined according to: i) the subtree's depth, ii) the number of nodes of the subtree, iii) the atomic weight of the subtree's root.

An example of representation for a Benzodiazepine is shown in Fig. 12 (for compound #4 in Table B.1 in the Appendix), while Fig. 13 shows the representation of a substituent (from compound #49 in Table B.1 in the Appendix).

5.2. Representation of Alkanes

We observe that the hydrogens suppressed graphs of Alkane molecules are trees and they can be represented as ordered rooted trees by the following minimal set of rules:

1. the carbon-hydrogens groups (C, CH, CH_2, CH_3) are associated with graph vertexes while bonds between carbon atoms are represented by edges;

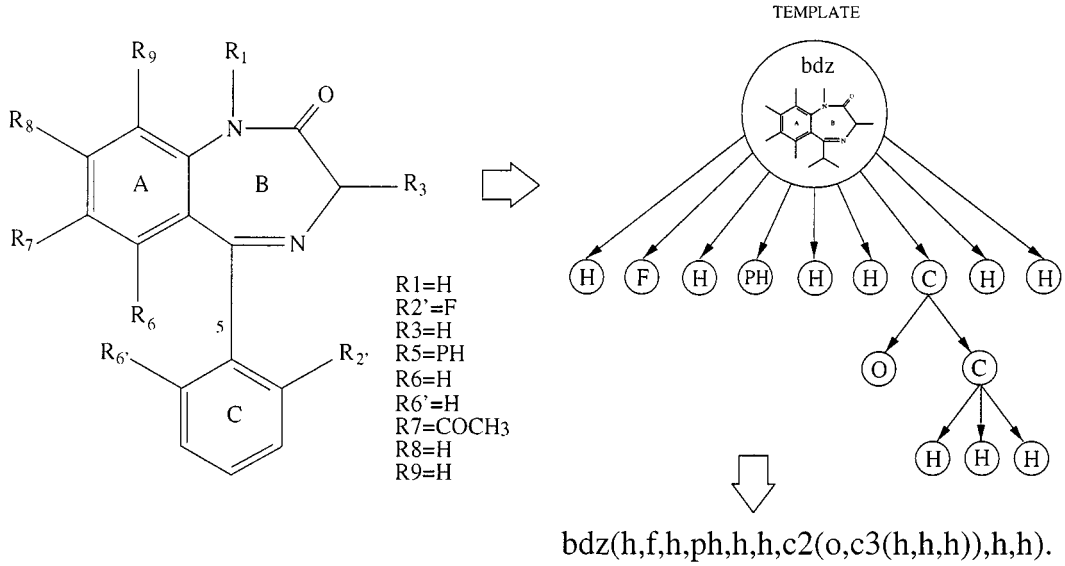


Figure 12. Example of representation for a benzodiazepine.

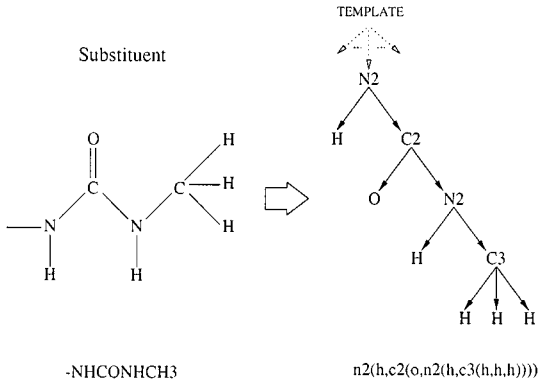


Figure 13. Example of representation of a substituent.

2. the root of the tree is defined as the first vertex of the main chain (i.e., the longest chain present in the compound) numbered from one end to the other according to I.U.P.A.C. rules (the direction is chosen so to assign the lowest numbers possible to side chains, resorting, when needed, to a lexicographic order); moreover, if there are two or more side chains in equivalent positions, instead of using the I.U.P.A.C. alphabetical order of the radicals names, we adopt an order based on the size of the side chains (i.e., depth of substructure);
3. the orientation of the edges follows the increasing levels of the trees;
4. the total order on the subtrees of each node is defined according to the depth of the substructure;

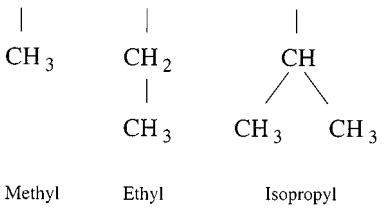


Figure 14. The three possible side chains occurring in our dataset.

we impose a total order on the three possible side chains occurring in our dataset: methyl $<$ ethyl $<$ isopropyl. The three radicals are shown in Fig. 14.

Examples of representations for Alkanes are shown in Fig. 15.

6. Experimental Results

In this section we report experimental results obtained by using the Sum of Square Errors as global error function. Due to the low number of training data and to avoid overfitting, several expedients were used for setting the Cascade Correlation for structure parameters. First of all, no connection between hidden units were allowed. Then the gains of the sigmoids of the hidden units were set to 0.4. Finally, an incremental strategy (*i-strategy*) on the number of training epochs was adopted for each new inserted hidden node. This was

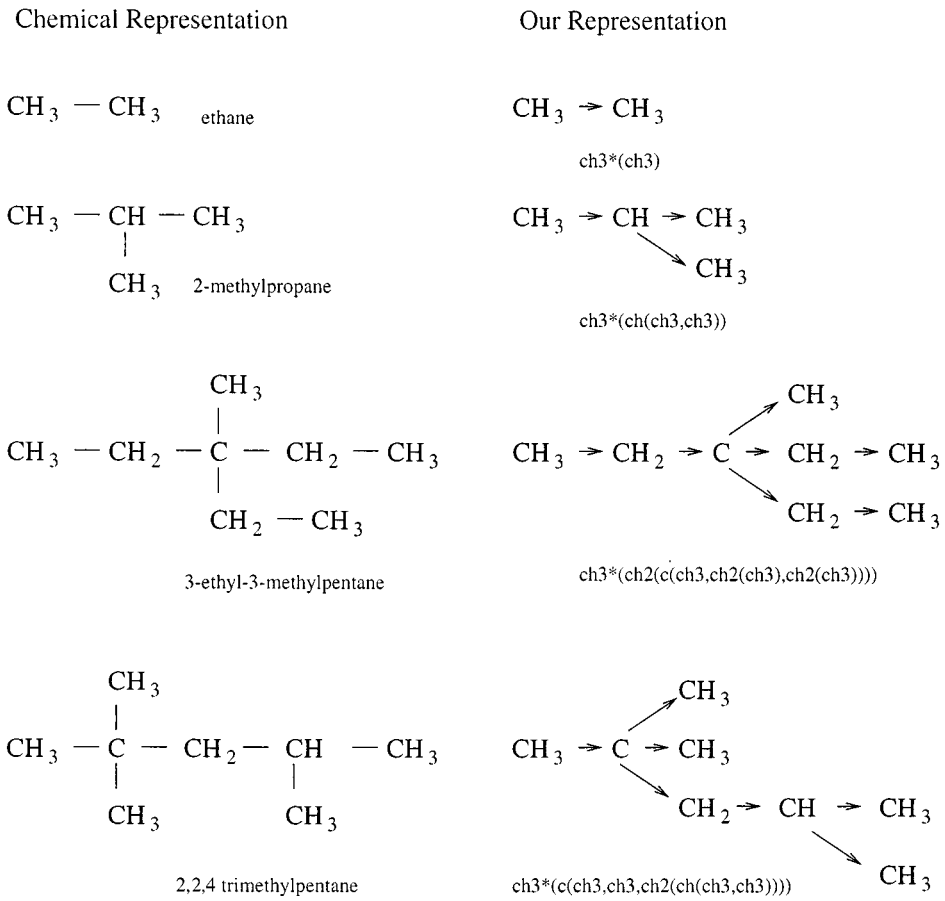


Figure 15. Examples of representations for Alkanes.

done because allowing few epochs to the first nodes decreases the probability of overfitting, by avoiding the increase of the weight values and the subsequent saturation of the units. On the other hand, lately introduced nodes, which work with small gradients due to the reduction of the residual error, take advantage from the increased number of epochs.

An initial set of preliminary trials were performed in order to determine an admissible range for the learning parameters. However, no effort was done to optimize these parameters.

6.1. Benzodiazepines

For the analysis of the data set described in Section 4.1, four different splittings in disjoint training and test sets of the data were used (Data set I, II, II, and IV, respectively). Specifically, the first test set (5 compounds) has been chosen as it contains the same compounds used by Hadjipavlou-Litina and Hansch for the validation of their treatment. The second data set is obtained from Data set I by removing 4 racemic compounds from the training set and one racemic compound from the test set. This allows the experimentation of our approach without the racemic compounds which are commonly recognized to introduce ambiguous information. The test set of Data set III (5 compounds) has been selected as it simultaneously shows a significant molecular diversity and a wide range of affinity values. Furthermore, the included compounds were selected so that substituents, already known to increase the affinity on given positions, appear in turn in place of H-atoms, which allows the decoupling of the effect of each substituent. So, a good generalization on this test set means that the network is able to capture the relevant aspects for the prediction. The test set of Data set IV (4 compounds) has been randomly chosen so to

Table 1. Results obtained for Benzodiazepines on training data set I by Hadjipavlou-Litina and Hansch (first row), by a "null model" (second row) and on all the training data sets by Cascade Correlation for Structures. Results obtained for different learning settings are reported for training data set II. The mean absolute error, the correlation coefficient (R) and the standard deviation of error (S) are reported.

Training set	#Units mean (min-max)	mean abs. error (min-max)	R	S
Hansch		0.311	0.847	0.390
Null model		0.580	0	0.702
Data set I	29.75 (23–40)	0.090(0.066–0.114)	0.99979	0.127
Data set II nis	48.0 (44–52)	0.110(0.099–0.120)	0.99973	0.144
Data set II is	15.3 (13–17)	0.100(0.076–0.114)	0.99978	0.130
Data set II tis	34.0 (27–38)	0.087(0.080–0.102)	0.99982	0.117
Data set III	19.7 (18–22)	0.087(0.072–0.105)	0.99985	0.098
Data set IV	16.5 (13–20)	0.099(0.078–0.132)	0.99976	0.131

test the sensitivity of the network to different learning conditions.

As target output for the networks we used $\log(1/C)$ normalized into the range $[0.6, 0.9]$. Concerning the label attached to each node, we use a bipolar localist representation to code (and to distinguish among) the types of chemical objects; in a bipolar localist representation each bit is assigned to one entity and it is equal to 1 if and only if the representation refers to that entity, otherwise it is set to -1 (bipolar): for example, the label for the F atom would be something like $[-1, -1, \ldots, -1, 1, -1, \ldots, -1, -1]$.

Six trials were carried out for the simulation involving each one of the different training sets. The initial connection weights used in each simulation were randomly set. Learning was stopped when the maximum error for a single compound was below 0.4 (which is actually 0.04 since we have scaled the target by a factor of 10). This tolerance is largely below the minimal tolerance needed for a correct classification of active drugs.

The main statistics computed over all the simulations for the training sets are reported in Table 1. Specifically, the results obtained by Hadjipavlou-Litina and Hansch, as well as the results obtained by the null model, i.e., the model where the expected mean value of the target is used to perform the prediction, are reported in the first and second row, respectively. For each data set, statistics on the number of inserted hidden units are reported for the Cascade Correlation for Structures network. The mean absolute error (Mean Abs. Error), the correlation coefficient (R) and the standard deviation of error (S), as defined in regression analysis, are reported in the last three columns. Note that Mean Abs. Error, R and S for Cascade Correlation for Structures are obtained by averaging over the performed trials (six trials); also the minimum and maximum values of the mean absolute error over these six trials are reported. For the Data set II we have reported the results obtained when using Cascade Correlation for Structures without i-strategy on the number of training epochs (nis), by using the i-strategy (is), and by an empirically tuned version of the i-strategy (tis).

The results for the corresponding test sets are reported in Table 2. In case of small test data sets the correlation coefficient is not meaningful so we prefer to report the maximum absolute error for the test data (Max Abs. Error), calculated as the mean over the six trials, and the corresponding minimum and maximum values of the maximum absolute error obtained for each trial.

In Figs. 16–19 we have plotted the output of the network versus the desired target for each splitting of the data.

Each point in the graphs represents the mean expected output, together with the deviation range, as computed over the six trials (i.e., the extremes of the deviation range correspond to the minimum and maximum output values computed over the six trials for each compound).

Table 2. Results obtained for Benzodiazepines on test data set I by Hadjipavlou-Litina and Hansch (first row), by a "null model" (second row) and on all the test data sets by Cascade Correlation for Structures. Results obtained for different learning settings are reported for test data set II. The mean absolute error, the mean of the maximum of the absolute error, and the standard deviation of error (S) are reported.

Test Set	Data #	Mean abs. error (min-max)	Mean max abs. error (min-max)	S
Hansch	5	1.272	1.750	1.307
Null model	5	1.239	1.631	1.266
Data set I	5	0.720 (0.611–0.792)	1.513 (1.106–1.654)	0.842
Data set II nis	4	0.757 (0.703–0.810)	0.991 (0.839–1.142)	0.792
Data set II is	4	0.662 (0.501–0.807)	0.839 (0.661–1.088)	0.683
Data set II tis	4	0.546 (0.444–0.653)	0.727 (0.523–0.973)	0.579
Data set III	5	0.255 (0.206–0.325)	0.606 (0.433–0.712)	0.329
Data set IV	4	0.379 (0.279–0.494)	0.746 (0.695–0.763)	0.460

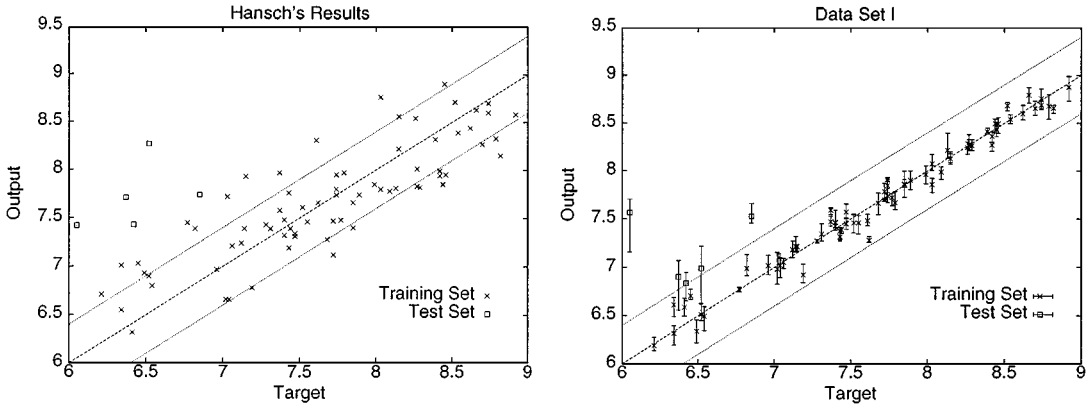


Figure 16. Output of the models proposed by Hadjipavlou-Litina and Hansch (left) and of the Cascade Correlation network (right) versus the desired target; both models use the same training and test sets (data set I). Each point in the right plot represents the mean expected output for Cascade Correlation network, together with the deviation range (minimum and maximum values), as computed over six trials. The tolerance region is shown on the plots. Note that in the plot to the right (CCS) the test data are located at the lower left corner.

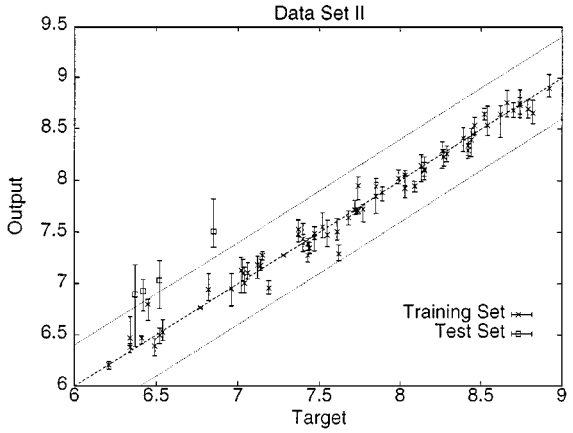


Figure 17. Results for data set II.

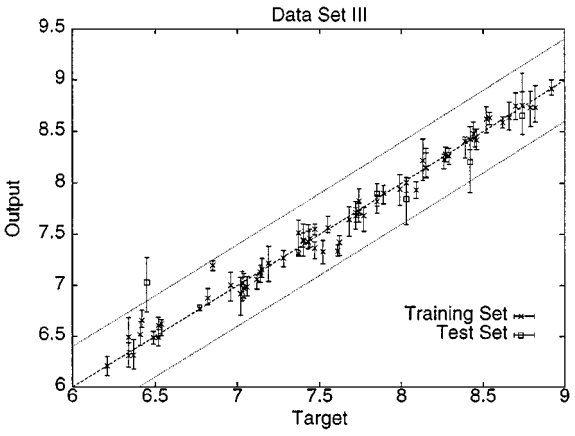


Figure 18. Data set III: The test data are spread across the input range.

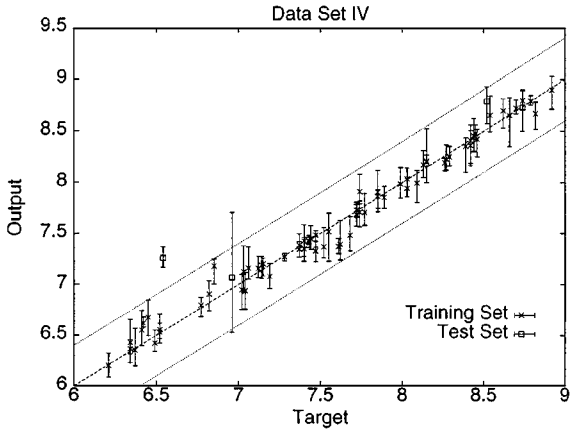


Figure 19. Data set IV: The large deviation range for the test compound with target close to 7 is explained by the presence of a substituent in the tested compound which does not occur in the training set.

6.2. Alkanes

As target output for the networks we used the boiling point in Celsius degrees normalized into the range $[-1.64, 1.74]$. Also in this case, a bipolar localist representation encoding the atom types was used.

For the sake of comparison, we tested the prediction ability using exactly the same 10-fold cross validation (15 compounds for each fold) used in [15]. Moreover, we repeated the procedure for four times. Learning was stopped when the maximum absolute error for a single compound was below 0.08.

The obtained results for the training data are reported in Table 3 and compared with the results obtained by different approaches, i.e., the results obtained by Cherqaoui et al. using 'ad hoc' Neural Networks, two different equations based on connectivity (χ) topological indices, and multilinear regression over the vectorial code for Alkanes. The results obtained on the test set are shown in Table 4 and compared with the MLP results obtained by Cherqaoui et al. It must be pointed out that the results are computed by removing the methane compound (#1 in Table A.1 in the Appendix) from the test set (for the MLP and CCS in Table 4), since it turns out to be an outlier. Particularly, from the point of view of our new approach that consider the structure of compounds, the methane (CH_4) is so structurally small that it does not represent a typical element in the class of Alkanes.

The training results for each fold of the Cross-Validation are reported in Table 5. In Fig. 20 the residual errors for each compound are reported. The data used for the plots are reported as well in the Appendix. Examples of training and test curves for two different instances of Cascade Correlation networks trained over the same fold, are shown in Fig. 21.

Table 3. Results obtained for Alkanes on training data set by Cascade Correlation for Structure (CCS), by Cherqaoui et al. using 'ad hoc' neural networks (MLP), by using topological indices and by using multi linear regression. The data are obtained by a 10-fold cross-validation with 15 compounds for each fold. The correlation coefficient (R) and the standard deviation of error (S) are reported.

Model	#Units	Mean abs. error	R	S
CCS (mean)	110.7	1.98	0.99987	2.51
Best MLP	7	2.22	0.99852	2.64
Top. Index 1			0.9916	6.36
Top. Index 2			0.9945	5.15
MLR			0.9917	6.51

Table 4. Results obtained for Alkanes on test data set by Cascade Correlation for Structure (CCS) and by 'ad hoc' neural networks (MLP). The data are obtained by a 10 fold cross-validation with 15 compounds for each fold. The last row of the table is computed over four different cross-validation evaluations.

Model	Mean abs. error	Max abs. error	R	S
Best MLP	3.01	10.42	0.9966	3.49
Best CCS	2.74	13.27	0.9966	3.5
Mean CCS	3.71	30.33	0.9917	5.43

Table 5. Results obtained for Alkanes on training data set by Cascade Correlation for Structures. The data are obtained by the 10 Cascade Correlation networks used for 10-fold cross-validation and averaged over 4 different trials. The last row reports the mean results, and it corresponds to the first row of Table 3.

Training set fold	#Units	Mean abs. error	R	S
1	115˜	2.42	0.99981	3.04
2	88˜	2.16	0.99984	2.80
3	110˜	1.89	0.99988	2.42
4	133˜	2.06	0.99986	2.60
5	99˜	1.60	0.99992	2.04
6	112˜	1.70	0.99990	2.24
7	83˜	2.41	0.99982	2.96
8	108˜	1.72	0.99991	2.09
9	149˜	1.81	0.99989	2.32
10	110˜	2.04	0.99986	2.61
Mean	110.7	1.98	0.99987	2.51

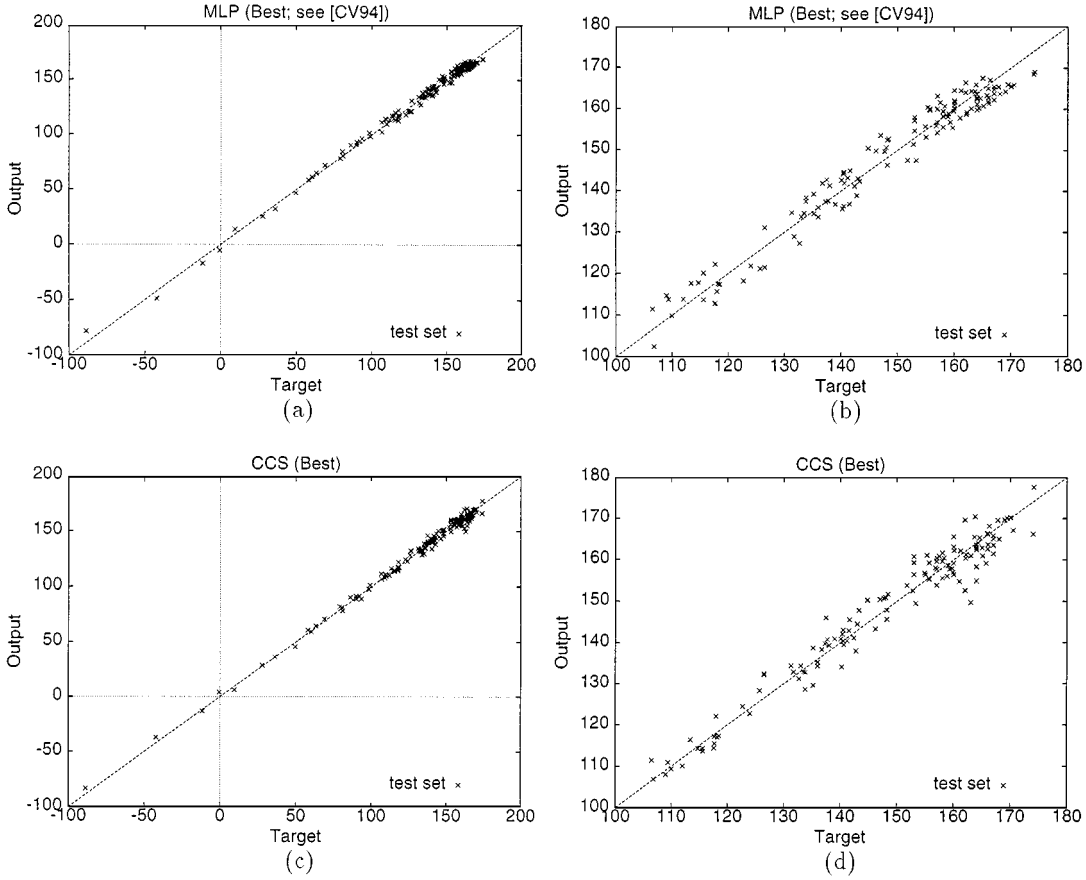


Figure 20. Outputs obtained by Cross-Validation by MLP (a) and CCS (c) versus the desired target for test data. Zooms in the range [100,180] of the plots are shown in (b) and (d), respectively. The values are expressed in °C.

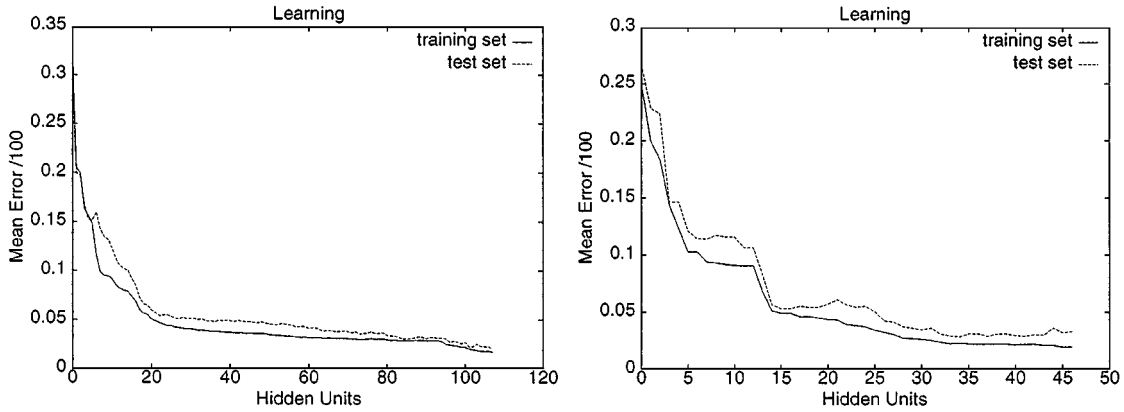


Figure 21. Mean training and test error for two different instances of Cascade Correlation networks trained over the same fold. The mean error is plotted versus the number of inserted hidden units.

7. Discussion

Concerning the evaluation of the performance of the proposed model for the treatment of Benzodiazepines, from the comparison with the results obtained by the traditional Hansch treatment, we can observe both a strong improvement in the fitting of the molecules included in the training set and in the test set. The experimental results suggest a relevant improvement over traditional QSAR techniques. In fact, this has been confirmed by the good results obtained for Data set III, where the compound for which predictions were worst is the one bearing hydrogen atoms in place of substituents which are relevant for the prediction (# 42 in Table B.1 in the Appendix). Moreover, the coherence of the proposed model has been confirmed by the good results obtained for Data set IV. Specifically, the compound which showed the maximum variance through the trials contains a substituent which never occurs in the training set, which explains the uncertainty in the response.

The results reported in Table 1 and Table 2 confirm the effectiveness of the *i-strategy* for learning. In fact, the improvements observed for Data Set II show a significant reduction both in the mean number of hidden units and in the mean absolute error for the test set. The improvement in generalization does not seem to be directly correlated with the mean number of hidden units, since when using the tuned version of the *i-strategy* (tis), the mean number of hidden units increases with respect to the standard *i-strategy* (is) while the generalization error decreases. This behavior, however, may be explained by the tuning procedure which may have invalidated the independence of the training set with respect to the test set.

The behavior of the model for the prediction of the boiling point of Alkanes demonstrates the ability of the model to be competitive with respect to 'ad hoc' techniques. In fact, the obtained results compares favorably versus the approach proposed by Cherqaoui et al. also in consideration of the fact that the vectorial representation of Alkanes retains the structural information which is known to be relevant to the prediction of the boiling point. Some comments concerning this comparison are however due. First of all, from the paper by Cherqaoui et al. only the best results are reported. Moreover, it is not clear whether the obtained results are obtained from just one trial of the 10-fold cross-validation procedure or from a set of several trials. For this reason, we have prefered to repeat the whole procedure 4 times and we have reported the cumulative results in row 3 of Table 4. These results show some variability in the output of the networks, demonstrating the strong dependence of the results from the initial setting of the weights and of the learning parameters. This variability has already been observed for other standard Cascade Correlation networks and it is mainly explained as a side effect of the several maxima which characterize the correlation function. Since it is well known that feedforward networks are more stable, it would be interesting to try the following procedure: Cascade Correlation networks are used to establish the mean number of hidden units needed to learn the training set; then this number of hidden units is used to define a set of feedforward neural networks for structure [7] which are then used to perform the prediction.

The above mentioned variability is also observed from the results reported in Table 5. Unfortunately, due to the lack of information, as already mentioned, it was not possible to compile a similar table for the standard networks used by Cherqaoui et al.

What we would like to stress here is that the experimental results seem to confirm that our approach can be used, without substantial modifications, both to QSAR and QSPR tasks, obtaining competitive or even better results with respect to traditional approaches.

8. Conclusion

We have demonstrated that the application of neural networks for structures to QSAR/QSPR tasks allows the treatment of different computational tasks by using the same basic representations for chemical compounds, obtaining improved prediction results with respect to traditional equational approaches for QSAR and competitive results with respect to 'ad hoc' designed representations and MLP networks in QSPR. It must be stressed that for QSAR, no physico-chemical descriptor was used by our model in this paper.

The main advantage of the proposed approach with respect to topological indices is that in our case no a priori definition of structural features is required. Specifically, since the learning phase involves both the encoding process and the regression one, the numerical encoding for the chemical structures devised by the encoding network are optimized with respect to the prediction task. Of course, this is not the case for topological indices which need to be devised and optimized through a trial and error procedure by experts in the fields of application. Moreover, in our approach it

is possible to store into the label attached to each node information at different levels of abstraction, such as the atom types or functional groups, allowing a flexible treatment of different aspects of the chemical functionality.

Concerning a comparison with respect to approaches based on feedforward networks, the main advantage resides in the fact that the encoding of chemical structures does not depend on a fixed vectorial or template based representation. In fact, due to the dynamical nature of the computational model, our approach is able to adapt the encoding process to the specific morphology of each single compound.

Moreover, the generality of the compound representations used by our approach allows the simultaneous treatment of chemically heterogeneous compounds. Finally, our approach must be regarded as a major step towards a fully structural representation and treatment of the chemical compounds using neural networks. This approach seems to be very promising since it is able to address computational problems which still are not reachable from traditional symbol-based systems, such as Inductive Logic Programming (ILP) [35]. In fact, even if ILP has been successfully applied to some problems in chemistry, the prediction of a numerical value from the structure of the compound does not seem to be possible within the ILP framework.

Appendix A: Dataset for Alkanes

1 ch3f(h).
2 ch3f(ch3).
3 ch3f(ch2(ch3)).
4 ch3f(ch(ch3,ch3)).
5 ch3f(ch2(ch2(ch3))).
6 ch3f(c(ch3,ch3,ch3)).
7 ch3f(ch(ch3,ch2(ch3))).
8 ch3f(ch2(ch2(ch2(ch3)))).
9 ch3f(c(ch3,ch3,ch2(ch3))).
10 ch3f(ch(ch3,ch(ch3,ch3))).
11 ch3f(ch(ch3,ch2(ch2(ch3)))).
12 ch3f(ch2(ch(ch3,ch2(ch3)))).
13 ch3f(ch2(ch2(ch2(ch2(ch3))))).
14 ch3f(c(ch3,ch3,ch(ch3,ch3))).
15 ch3f(c(ch3,ch3,ch2(ch2(ch3)))).
16 ch3f(ch2(c(ch3,ch3,ch2(ch3)))).
17 ch3f(ch(ch3,ch(ch3,ch2(ch3)))).
18 ch3f(ch(ch3,ch2(ch(ch3,ch3)))).
19 ch3f(ch(ch3,ch2(ch2(ch2(ch3))))).
20 ch3f(ch2(ch(ch3,ch2(ch2(ch3))))).
21 ch3f(ch2(ch(ch2(ch3),ch2(ch3)))).
22 ch3f(ch2(ch2(ch2(ch2(ch2(ch3)))))).
23 ch3f(c(ch3,ch3,c(ch3,ch3,ch3))).
24 ch3f(c(ch3,ch3,ch(ch3,ch2(ch3)))).
25 ch3f(ch(ch3,c(ch3,ch3,ch2(ch3)))).
26 ch3f(c(ch3,ch3,ch2(ch(ch3,ch3)))).
27 ch3f(c(ch3,ch3,ch2(ch2(ch2(ch3))))).
28 ch3f(ch2(c(ch3,ch3,ch2(ch2(ch3))))).
29 ch3f(ch2(c(ch3,ch2(ch3),ch2(ch3)))).
30 ch3f(ch(ch3,ch(ch3,ch(ch3,ch3)))).
31 ch3f(ch(ch3,ch(ch3,ch2(ch2(ch3))))).
32 ch3f(ch(ch3,ch(ch2(ch3),ch2(ch3)))).
33 ch3f(ch2(ch(ch3,ch(ch3,ch2(ch3))))).
34 ch3f(ch(ch3,ch2(ch(ch3,ch2(ch3))))).
35 ch3f(ch(ch3,ch2(ch2(ch(ch3,ch3))))).
36 ch3f(ch(ch3,ch2(ch2(ch2(ch2(ch3)))))).
37 ch3f(ch2(ch(ch3,ch2(ch2(ch2(ch3)))))).
38 ch3f(ch2(ch2(ch(ch3,ch2(ch2(ch3)))))).
39 ch3f(ch2(ch(ch2(ch3),ch2(ch2(ch3))))).
40 ch3f(ch2(ch2(ch2(ch2(ch2(ch2(ch3)))))).
41 ch3f(c(ch3,ch3,c(ch3,ch3,ch2(ch3)))).
42 ch3f(c(ch3,ch3,ch(ch3,ch(ch3,ch3)))).
43 ch3f(c(ch3,ch3,ch(ch3,ch2(ch2(ch3))))).
44 ch3f(c(ch3,ch3,ch(ch2(ch3),ch2(ch3)))).
45 ch3f(ch2(c(ch3,ch3,ch(ch3,ch2(ch3))))).
46 ch3f(ch(ch3,c(ch3,ch3,ch(ch3,ch3)))).
47 ch3f(ch(ch3,c(ch3,ch3,ch2(ch2(ch3))))).
48 ch3f(ch(ch3,c(ch3,ch2(ch3),ch2(ch3)))).
49 ch3f(c(ch3,ch3,ch2(c(ch3,ch3,ch3)))).
50 ch3f(c(ch3,ch3,ch2(ch(ch3,ch2(ch3))))).
51 ch3f(ch(ch3,ch2(c(ch3,ch3,ch2(ch3))))).
52 ch3f(c(ch3,ch3,ch2(ch2(ch(ch3,ch3))))).
53 ch3f(c(ch3,ch3,ch2(ch2(ch2(ch2(ch3)))))).
54 ch3f(ch2(c(ch3,ch3,ch2(ch2(ch2(ch3)))))).
55 ch3f(ch2(ch2(c(ch3,ch3,ch2(ch2(ch3)))))).
56 ch3f(ch2(c(ch3,ch2(ch3),ch2(ch2(ch3))))).
57 ch3f(ch2(c(ch2(ch3),ch2(ch3),ch2(ch3)))).
58 ch3f(ch(ch3,ch(ch3,ch(ch3,ch2(ch3))))).
59 ch3f(ch(ch3,ch(ch2(ch3),ch(ch3,ch3)))).
60 ch3f(ch(ch3,ch(ch3,ch2(ch(ch3,ch3))))).
61 ch3f(ch(ch3,ch(ch3,ch2(ch2(ch2(ch3)))))).
62 ch3f(ch(ch3,ch(ch2(ch3),ch2(ch2(ch3))))).
63 ch3f(ch2(ch(ch3,ch(ch3,ch2(ch2(ch3)))))).
64 ch3f(ch2(ch(ch2(ch3),ch(ch3,ch2(ch3))))).
65 ch3f(ch(ch3,ch2(ch(ch3,ch2(ch2(ch3)))))).
66 ch3f(ch(ch3,ch2(ch(ch2(ch3),ch2(ch3))))).
67 ch3f(ch2(ch(ch3,ch2(ch(ch3,ch2(ch3)))))).
68 ch3f(ch(ch3,ch2(ch2(ch(ch3,ch2(ch3)))))).
69 ch3f(ch(ch3,ch2(ch2(ch2(ch(ch3,ch3)))))).
70 ch3f(ch(ch3,ch2(ch2(ch2(ch2(ch2(ch3)))))).
71 ch3f(ch2(ch(ch3,ch2(ch2(ch2(ch2(ch3)))))).
72 ch3f(ch2(ch2(ch(ch3,ch2(ch2(ch2(ch3)))))).
73 ch3f(ch2(ch(ch2(ch3),ch2(ch2(ch2(ch3)))))).
74 ch3f(ch2(ch2(ch(ch2(ch3),ch2(ch2(ch3)))))).
75 ch3f(ch2(ch2(ch2(ch2(ch2(ch2(ch2(ch3)))))))).
76 ch3f(c(ch3,ch3,c(ch3,ch3,ch(ch3,ch3)))).
77 ch3f(c(ch3,ch3,c(ch3,ch3,ch2(ch2(ch3))))).

78 ch3f(c(ch3,ch3,c(ch3,ch2(ch3),ch2(ch3)))).
79 ch3f(ch2(c(ch3,ch3,c(ch3,ch3,ch2(ch3))))).
80 ch3f(c(ch3,ch3,ch(ch3,c(ch3,ch3,ch3)))).
81 ch3f(c(ch3,ch3,ch(ch3,ch(ch3,ch2(ch3))))).
82 ch3f(c(ch3,ch3,ch(ch2(ch3),ch(ch3 ,ch3)))).
83 ch3f(ch(ch3,ch(ch3,c(ch3,ch3,ch2(ch3))))).
84 ch3f(c(ch3,ch3,ch(ch3,ch2(ch(ch3,ch3))))).
85 ch3f(c(ch3,ch3,ch(ch3,ch2(ch2(ch2(ch3)))))).
86 ch3f(c(ch3,ch3,ch(ch2(ch3),ch2(ch2(ch3))))).
87 ch3f(ch2(c(ch3,ch3,ch(ch3,ch2(ch2(ch3)))))).
88 ch3f(ch2(c(ch3,ch3,ch(ch2(ch3),ch2(ch3))))).
89 ch3f(ch(ch3,c(ch3,ch3,ch(ch3,ch2(ch3))))).
90 ch3f(ch2(ch(ch3,c(ch3,ch3,ch2(ch2(ch3)))))).
91 ch3f(ch2(c(ch3,ch2(ch3),ch(ch3,ch2(ch3))))).
92 ch3f(ch(ch3,c(ch3,ch2(ch3),ch(ch3,ch3)))).
93 ch3f(ch(ch3,c(ch3,ch3,ch2(ch(ch3,ch3))))).
94 ch3f(ch(ch3,c(ch3,ch3,ch2(ch2(ch2(ch3)))))).
95 ch3f(ch(ch3,c(ch3,ch2(ch3),ch2(ch2(ch3))))).
96 ch3f(ch(ch3,c(ch2(ch3),ch2(ch3),ch2(ch3)))).
97 ch3f(c(ch3,ch3,ch2(c(ch3,ch3,ch2(ch3))))).
98 ch3f(c(ch3,ch3,ch2(ch(ch3 ,ch(ch3,ch3))))).
99 ch3f(c(ch3,ch3,ch2(ch(ch3,ch2(ch2(ch3)))))).
100 ch3f(c(ch3,ch3,ch2(ch(ch2(ch3),ch2(ch3))))).
101 ch3f(ch2(c(ch3,ch3,ch2(ch(ch3,ch2(ch3)))))).
102 ch3f(ch(ch3,ch2(c(ch3,ch3,ch2(ch2(ch3)))))).
103 ch3f(ch(ch3,ch2(c(ch3,ch2(ch3),ch2(ch3))))).
104 ch3f(c(ch3,ch3,ch2(ch2(c(ch3,ch3,ch3))))).
105 ch3f(c(ch3,ch3,ch2(ch2(ch(ch3,ch2(ch3)))))).
106 ch3f(ch(ch3,ch2(ch2(c(ch3,ch3,ch2(ch3)))))).
107 ch3f(c(ch3,ch3,ch2(ch2(ch2(ch(ch3,ch3)))))).
108 ch3f(c(ch3,ch3,ch2(ch2(ch2(ch2(ch2(ch3))))))).
109 ch3f(ch2(c(ch3,ch3,ch2(ch2(ch2(ch2(ch3))))))).
110 ch3f(ch2(ch2(c(ch3,ch3,ch2(ch2(ch2(ch3))))))).
111 ch3f(ch2(c(ch3,ch2(ch3),ch2(ch2(ch2(ch3)))))).
112 ch3f(ch2(ch2(c(ch3,ch2(ch3),ch2(ch2(ch3)))))).
113 ch3f(ch2(c(ch2(ch3),ch2(ch3),ch2(ch2(ch3))))).
114 ch3f(ch(ch3,ch(ch3,ch(ch3,ch(ch3,ch3))))).
115 ch3f(ch(ch3,ch(ch3,ch(ch3,ch2(ch2(ch3)))))).
116 ch3f(ch(ch3,ch(ch3,ch(ch2(ch3),ch2(ch3))))).
117 ch3f(ch(ch3,ch(ch2(ch3),ch(ch3,ch2(ch3))))).
118 ch3f(ch2(ch(ch3,ch(ch3,ch(ch3,ch2(ch3)))))).
119 ch3f(ch(ch3,ch(ch(ch3,ch3),ch(ch3,ch3)))).
120 ch3f(ch(ch3,ch(ch(ch3,ch3),ch2(ch2(ch3))))).
121 ch3f(ch(ch3,ch(ch3,ch2(ch(ch3,ch2(ch3)))))).
122 ch3f(ch(ch3,ch(ch2(ch3),ch2(ch(ch3,ch3))))).
123 ch3f(ch(ch3,ch2(ch(ch3,ch(ch3,ch2(ch3)))))).
124 ch3f(ch(ch3,ch(ch3,ch2(ch2(ch(ch3,ch3)))))).
125 ch3f(ch(ch3,ch(ch3,ch2(ch2(ch2(ch2(ch3)))))).
126 ch3f(ch(ch3,ch(ch2(ch3),ch2(ch2(ch2(ch3)))))).
127 ch3f(ch2(ch(ch3,ch(ch3,ch2(ch2(ch2(ch3)))))).
128 ch3f(ch2(ch2(ch(ch(ch3,ch3),ch2(ch2(ch3)))))).
129 ch3f(ch2(ch(ch3,ch(ch2(ch3),ch2(ch2(ch3)))))).
130 ch3f(ch2(ch2(ch(ch3,ch(ch3,ch2(ch2(ch3)))))).
131 ch3f(ch2(ch(ch2(ch3),ch(ch3,ch2(ch2(ch3)))))).
132 ch3f(ch2(ch(ch2(ch3),ch(ch2(ch3),ch2(ch3))))).
133 ch3f(ch(ch3,ch2(ch(ch3,ch2(ch(ch3,ch3)))))).
134 ch3f(ch(ch3,ch2(ch(ch3,ch2(ch2(ch2(ch3)))))).
135 ch3f(ch(ch3,ch2(ch(ch2(ch3),ch2(ch2(ch3)))))).
136 ch3f(ch2(ch(ch3,ch2(ch(ch3,ch2(ch2(ch3)))))).
137 ch3f(ch2(ch(ch2(ch3),ch2(ch(ch3,ch2(ch3)))))).
138 ch3f(ch(ch3,ch2(ch2(ch(ch3,ch2(ch2(ch3)))))).
139 ch3f(ch(ch3,ch2(ch2(ch(ch2(ch3),ch2(ch3)))))).
140 ch3f(ch2(ch(ch3,ch2(ch2(ch(ch3,ch2(ch3)))))).
141 ch3f(ch(ch3,ch2(ch2(ch2(ch(ch3,ch2(ch3)))))).
142 ch3f(ch(ch3,ch2(ch2(ch2(ch2(ch(ch3,ch3)))))).
143 ch3f(ch(ch3,ch2(ch2(ch2(ch2(ch2(ch2(ch3))))))).
144 ch3f(ch2(ch(ch3,ch2(ch2(ch2(ch2(ch2(ch3))))))).
145 ch3f(ch2(ch2(ch(ch3,ch2(ch2(ch2(ch2(ch3))))))).
146 ch3f(ch2(ch(ch2(ch3),ch2(ch2(ch2(ch2(ch3)))))).
147 ch3f(ch2(ch2(ch2(ch(ch3,ch2(ch2(ch2(ch3))))))).
148 ch3f(ch2(ch2(ch(ch2(ch3),ch2(ch2(ch2(ch3)))))).
149 ch3f(ch2(ch2(ch(ch2(ch2(ch3)),ch2(ch2(ch3)))))).
150 ch3f(ch2(ch2(ch2(ch2(ch2(ch2(ch2(ch2(ch3))))))))).

Table A.1. Test data for alkanes.

#	Alkane	Target (°C)	Output Cherqaoui	Error Cherqaoui	Output CCS	Error CCS
1	methane	−164	−149.55	−14.45	−79.76	−84.24
2	ethane	−88.6	−78.18	−10.42	−83.02	−5.58
3	propane	−42.1	−48.65	6.55	−36.90	−5.20
4	2-methylpropane	−11.7	−16.91	5.21	−12.89	1.19
5	butane	−0.5	−5.09	4.59	4.07	−4.57
6	2,2-dimethylpropane	9.5	13.92	−4.42	6.17	3.33
7	2-methylbutane	27.8	25.64	2.16	28.86	−1.06
8	pentane	36.1	32.51	3.59	36.60	−0.50

(*Continued on next page*.)

Table A.1. (*continued*).

#	Alkane	Target (°C)	Output Cherqaoui	Error Cherqaoui	Output CCS	Error CCS
9	2,2-dimethylbutane	49.7	47.06	2.64	45.80	3.90
10	2,3-dimethylbutane	58	58.76	−0.76	60.70	−2.70
11	2-methylpentane	60.3	61.77	−1.47	59.31	0.99
12	3-methylpentane	63.3	65.21	−1.91	64.42	−1.12
13	hexane	69	72.1	−3.1	70.87	−1.87
14	2,2,3-trimethylbutane	80.9	80.79	0.11	78.18	2.72
15	2,2-dimethylpentane	79.2	78.41	0.79	82.00	−2.80
16	3,3-dimethylpentane	86.1	90.7	−4.6	90.27	−4.17
17	2,3-dimethylpentane	89.8	90.99	−1.19	91.38	−1.58
18	2,4-dimatilpentane	80.5	84.93	−4.43	81.04	−0.54
19	2-methylhexane	90	93.3	−3.3	88.94	1.06
20	3-methylhexane	92	94.3	−2.3	91.34	0.66
21	3-ethylpentane	93.5	96.53	−3.03	88.90	4.60
22	heptane	98.4	101.83	−3.43	97.68	0.72
23	2,2,3,3-tetramethylbutane	106.5	111.46	−4.96	111.45	−4.95
24	2,2,3-trimethylpentane	110	109.83	0.17	109.40	0.60
25	2,3,3-trimethylpentane	114.7	117.75	−3.05	114.26	0.44
26	2,2,4-trimethylpentane	99.2	98.47	0.73	101.28	−2.08
27	2,2-dimethylhexane	106.8	102.45	4.35	106.86	−0.06
28	3,3-dimethylhexane	112	113.83	−1.83	110.11	1.89
29	3-ethyl-3-methylpentane	118.2	117.5	0.7	116.99	1.21
30	2,3,4-trimethylpentane	113.4	117.63	−4.23	116.41	−3.01
31	2,3-dimethylhexane	115.6	113.71	1.89	113.64	1.96
32	3-ethyl-2-methylpentane	115.6	120.1	−4.5	114.34	1.26
33	3,4-dimethylhexane	117.7	122.19	−4.49	115.45	2.25
34	2,4-dimethylhexane	109.4	113.77	−4.37	110.89	−1.49
35	2,5-dimethylhexane	109	114.68	−5.68	108.04	0.96
36	2-methylheptane	117.6	112.79	4.81	114.38	3.22
37	3-methylheptane	118	115.65	2.35	122.07	−4.07
38	4-methylheptane	117.7	112.91	4.79	117.22	0.48
39	3-ethylhexane	118.5	117.42	1.08	117.36	1.14
40	octane	125.7	121.14	4.56	128.32	−2.62
41	2,2,3,3-tetramethylpentane	140.27	135.71	4.56	134.11	6.16
42	2,2,3,4-tetramethylpentane	133	133.74	−0.74	134.38	−1.38
43	2,2,3-trimethylhexane	131.7	128.96	2.74	132.88	−1.18
44	2,2-dimethyl-3-ethylpentane	133.83	138.39	−4.56	132.82	1.01
45	3,3,4-trimethylhexane	140.5	144.61	−4.11	143.08	−2.58
46	2,3,3,4-tetramethylpentane	141.5	136.83	4.67	142.93	−1.43
47	2,3,3-trimethylhexane	137.7	137.63	0.07	140.82	−3.12
48	2,3-dimethyl-3-ethylpentane	141.6	144.96	−3.36	145.57	−3.97
49	2,2,4,4-tetramethylpentane	122.7	118.23	4.47	124.51	−1.81
50	2,2,4-trimethylhexane	126.5	121.51	4.99	132.45	−5.95

(*Continued on next page*.)

Table A.1. (*continued*).

#	Alkane	Target (°C)	Output Cherqaoui	Error Cherqaoui	Output CCS	Error CCS
51	2,4,4-trimethylhexane	126.5	131.06	−4.56	132.20	−5.70
52	2,2,5-trimethylhexane	124	121.85	2.15	122.77	1.23
53	2,2-dimethylheptane	132.7	127.3	5.4	131.25	1.45
54	3,3-dimethylheptane	137.3	137.41	−0.11	139.78	−2.48
55	4,4-dimethylheptane	135.2	139.29	−4.09	138.75	−3.55
56	3-ethyl-3-methylhexane	140.6	141.93	−1.33	139.54	1.06
57	3,3-dimethylpentane	146.2	149.84	−3.64	143.36	2.85
58	2,3,4-trimethylhexane	139	136.81	2.19	140.97	−1.97
59	2,4-dimethyl-3-ethylpentane	136.73	141.92	−5.19	138.41	−1.68
60	2,3,5-trimethylhexane	131.3	134.71	−3.41	134.41	−3.11
61	2,3-dimethylpentane	140.5	136.29	4.21	140.45	0.05
62	3-ethyl-2-methylhexane	138	141.32	−3.32	139.42	−1.42
63	3,4-dimethylheptane	140.1	142.66	−2.56	140.92	−0.82
64	3-ethyl-4-methylhexane	140.4	144.21	−3.81	142.11	−1.71
65	2,4-dimethylheptane	133.5	134.6	−1.1	132.87	0.63
66	4-ethyl-2-methylhexane	133.8	137.5	−3.7	128.64	5.16
67	3,5-dimethylheptane	136	135.99	0.01	135.25	0.75
68	2,5-dimethylheptane	136	133.68	2.32	134.31	1.69
69	2,6-dimethylheptane	135.2	134.5	0.7	129.61	5.59
70	2-methyloctane	142.8	138.93	3.87	138.03	4.77
71	3-methyloctane	143.3	142.41	0.89	147.84	−4.54
72	4-methyloctane	142.4	141.19	1.21	141.21	1.19
73	3-ethylheptane	143	143.06	−0.06	144.55	−1.54
74	4-ethylheptane	141.2	143.24	−2.04	140.53	0.67
75	nonane	151.77	147.54	4.23	153.83	−2.06
76	2,2,3,3,4-pentamethylpentane	166.05	161.33	4.72	163.64	2.41
77	2,2,3,3-tetramethylhexane	158	159.44	−1.44	159.62	−1.62
78	3-ethyl-2,2,3-trimethylpentane	168	163.73	4.27	164.95	3.05
79	3,3,4,4-tetramethylhexane	170.5	165.88	4.62	167.11	3.39
80	2,2,3,4,4-pentamethylpentane	159.29	156.85	2.44	157.74	1.55
81	2,2,3,4-tetramethylhexane	154.9	155.77	−0.87	156.80	−1.90
82	3-ethyl-2,2,4-trimethylpentane	155.3	160.43	−5.13	161.09	−5.79
83	2,3,4,4-tetramethylhexane	162.2	159.22	2.98	161.24	0.96
84	2,2,3,5-tetramethylhexane	148.4	152.73	−4.33	151.69	−3.29
85	2,2,3-trimethylheptane	158	155.62	2.38	155.55	2.45
86	2,2-dimethyl-3-ethylhexane	159	158.89	0.11	158.64	0.37
87	3,3,4-trimethylheptane	164	163.6	0.4	162.91	1.09
88	3,3-dimethyl-4-ethylhexane	165	167.53	−2.53	162.87	2.13
89	2,3,3,4-tetramethylhexane	164.59	162.62	1.97	165.28	−0.69
90	3,4,4-trimethylheptane	164	162.76	1.24	162.44	1.56
91	3,4-dimethyl-3-ethylhexane	170	165.52	4.48	170.22	−0.22
92	3-ethyl-2,3,4-trimethylpentane	169.44	165.95	3.49	170.08	−0.64

(*Continued on next page*.)

Table A.1. (*continued*).

#	Alkane	Target (°C)	Output Cherqaoui	Error Cherqaoui	Output CCS	Error CCS
93	2,3,3,5-tetramethylhexane	153	157.2	−4.2	160.82	−7.82
94	2,3,3-trimethylheptane	160.1	161.7	−1.6	162.24	−2.14
95	2,3-dimethyl-3-ethylhexane	169	164.25	4.75	169.61	−0.61
96	3,3-diethyl-2-methylpentane	174	168.52	5.48	166.18	7.82
97	2,2,4,4-tetramethylhexane	153.3	147.55	5.75	149.53	3.78
98	2,2,4,5-tetramethylhexane	148.2	146.37	1.83	147.91	0.29
99	2,2,4-trimethylheptane	147.7	149.67	−1.97	150.75	−3.05
100	2,2-dimethyl-4-ethylhexane	147	153.59	−6.59	150.47	−3.47
101	3,3,5-trimethylheptane	155.68	160.05	−4.37	159.24	−3.56
102	2,4,4-trimethylheptane	153	154.7	−1.7	156.43	−3.43
103	2,4-dimethyl-4-ethylhexane	158	158.32	−0.32	161.66	−3.66
104	2,2,5,5-tetramethylhexane	137.46	142.79	−5.33	146.02	−8.56
105	2,2,5-trimethylheptane	148	150.67	−2.67	150.89	−2.89
106	2,5,5-trimethylheptane	152.8	151.43	1.37	152.52	0.28
107	2,2,6-trimethylheptane	148.2	152.51	−4.31	145.66	2.54
108	2,2-dimethyloctane	155	153.2	1.8	156.39	−1.38
109	3,3-dimethyloctane	161.2	164.51	−3.31	162.18	−0.98
110	4,4-dimethyloctane	157.5	161.61	−4.11	160.88	−3.38
111	3-ethyl-3-methylheptane	163.8	165.93	−2.13	170.45	−6.65
112	4-ethyl-4-methylheptane	167	162.25	4.75	161.46	5.54
113	3,3-diethylhexane	166.3	167.08	−0.78	168.14	−1.84
114	2,3,4,5-tetramethylhexane	161	157.85	3.15	154.78	6.22
115	2,3,4-trimethylheptane	163	160.12	2.88	160.94	2.06
116	2,3-dimethyl-4-ethylhexane	164	161.23	2.77	158.32	5.68
117	2,4-dimethyl-3-ethylhexane	164	162.46	1.54	163.55	0.45
118	3,4,5-trimethylheptane	164	159.79	4.21	154.91	9.09
119	2,4-dimethyl-3-isopropylpentane	157.04	163.2	−6.16	160.43	−3.39
120	3-isopropyl-2-methylhexane	163	164.41	−1.41	149.73	13.28
121	2,3,5-trimethylheptane	157	156.77	0.23	157.74	−0.74
122	2,5-dimethyl-3-ethylihexane	157	160.03	−3.03	159.63	−2.63
123	2,4,5-trimethylheptane	157	154.27	2.73	153.88	3.12
124	2,3,6-trimethylheptane	155.7	159.78	−4.08	155.30	0.40
125	2,3-dimethyloctane	164.31	160.22	4.09	164.07	0.24
126	3-ethyl-2-methylheptane	166	161.84	4.16	166.36	−0.36
127	3,4-dimethyloctane	166	162.77	3.23	162.56	3.44
128	4-isopropylheptane	160	164.51	−4.51	159.17	0.83
129	4-ethyl-3-methylheptane	167	164.82	2.18	163.51	3.49
130	4,5-dimethyloctane	162.1	158.76	3.34	160.37	1.73
131	3-ethyl-4-methylheptane	167	165.42	1.58	165.04	1.96
132	3,4-diethylhexane	162	166.43	−4.43	169.63	−7.63
133	2,4,6-trimethylheptane	144.8	150.51	−5.71	150.29	−5.48
134	2,4-dimethyloctane	153	157.98	−4.98	159.29	−6.29
135	4-ethyl-2-methylheptane	160	160.25	−0.25	165.53	−5.53

(*Continued on next page*.)

Table A.1. (*continued*).

#	Alkane	Target (°C)	Output Cherqaoui	Error Cherqaoui	Output CCS	Error CCS
136	3,5-dimethyloctane	160	161.02	−1.02	163.10	−3.10
137	3-ethyl-5-methylheptane	158.3	160.71	−2.41	160.43	−2.13
138	2,5-dimethyloctane	156.8	155.98	0.82	158.06	−1.26
139	5-ethyl-2-methylheptane	159.7	155.51	4.19	157.90	1.80
140	3,6-dimethyloctane	160	162.05	−2.05	156.39	3.61
141	2,6-dimethyloctane	158.84	158.13	0.71	156.00	2.84
142	2,7-dimethyloctane	159.87	159.55	0.32	156.91	2.96
143	2-methylnonane	167	163.78	3.22	165.80	1.20
144	3-methylnonane	167.8	165.38	2.42	169.55	−1.75
145	4-methylnonane	165.7	163.34	2.36	159.17	6.53
146	3-ethyloctane	166	165.05	0.95	166.23	−0.23
147	5-methylnonane	165.1	160.74	4.36	160.94	4.16
148	4-ethyloctane	163.64	164.27	−0.63	165.48	−1.84
149	4-propilheptane	162	164.06	−2.06	152.62	9.38
150	decane	174.12	168.99	5.13	177.53	−3.41

Appendix B: Dataset for Benzodiazepines

1 bzd(c3(h,h,h),f,h,ph,h,h,c1(n),h,h).
2 bzd(h,h,h,ph,h,h,c2(h,c2(h,h)),h,h).
3 bzd(h,f,h,ph,h,h,h,h,h).
4 bzd(h,f,h,ph,h,h,c2(o,c3(h,h,h)),h,h).
5 bzd(h,h,h,ph,h,h,c3(f,f,f),h,h).
6 bzd(c3(h,h,h),h,h,ph,h,h,cl,h,h).
7 bzd(c3(h,h,h),cl,h,ph,h,cl,cl,h,h).
8 bzd(h,f,h,ph,h,h,n1(n1(n)),h,h).
9 bzd(c3(h,h,h),f,h,ph,h,h,n2(o,o),h,h).
10 bzd(h,c3(f,f,f),h,ph,h,h,n2(o,o),h,h).
11 bzd(c3(h,h,h),f,h,ph,h,h,i,h,h).
12 bzd(c3(h,h,h),f,h,ph,h,f,br,h,h).
13 bzd(h,f,h,ph,h,h,cl,h,h).
14 bzd(h,cl,h,ph,h,h,cl,h,h).
15 bzd(h,cl,h,ph,h,h,n2(o,o),h,h).
16 bzd(h,f,h,ph,h,h,n2(o,o),h,h).
17 bzd(c3(h,h,h),f,h,ph,h,h,f,h,h).
18 bzd(c3(h,h,h),h,h,ph,h,h,f,h,h).
19 bzd(h,f,h,ph,h,h,f,h,h).
20 bzd(h,h,h,ph,h,h,cl,h,h).
21 bzd(h,f,h,ph,h,f,cl,h,h).
22 bzd(c3(h,h,h),f,h,ph,h,f,cl,h,h).
23 bzd(h,cl,h,ph,h,f,cl,h,h).
24 bzd(h,cl,h,ph,h,cl,cl,h,h).
25 bzd(h,h,h,ph,h,h,n2(o,o),h,h).
26 bzd(c3(h,h,h),cl,h,ph,h,h,n2(o,o),h,h).
27 bzd(c3(h,h,c3(h,h,o1(h))),f,h,ph,h,h,cl,h,h).
28 bzd(h,f,c3(h,h,h),ph,h,h,cl,h,h).
29 bzd(h,cl,c3(h,h,h),ph,h,h,n2(o,o),h,h).
30 bzd(c3(h,h,h),f,c3(h,h,h),ph,h,h,n2(o,o),h,h).
31 bzd(h,h,h,py,h,h,br,h,h).
32 bzdtg(h,cl,h,ph,h,h,cl,h,h).
33 bzd(h,f,h,ph,h,f,h,h,h).
34 bzd(c3(h,h,h),f,h,ph,h,h,h,h,h).
35 bzd(c3(h,h,h),cl,h,ph,h,h,h,h,h).
36 bzd(h,f,h,ph,h,f,h,cl,h).
37 bzd(h,f,h,ph,h,h,h,c3(h,h,h),h).
38 bzd(h,f,h,ph,h,h,cl,cl,h).
39 bzd(h,f,h,ph,h,h,c3(h,h,h),cl,h).
40 bzd(c3(h,h,h),h,h,ph,h,h,n2(h,h),h,h).
41 bzd(h,h,h,ph,h,h,n2(h,h),h,h).
42 bzd(h,h,h,ph,h,h,h,h,h).
43 bzd(c3(h,h,h),f,h,ph,h,h,n2(h,o1(h)),h,h).
44 bzd(h,cl,h,ph,h,h,n2(h,h),h,h).
45 bzd(h,h,h,ph,h,h,c2(h,o),h,h).
46 bzd(h,h,h,ph,h,h,f,h,h).
47 bzd(h,h,h,ph,h,h,c3(h,h,c3(h,h,h)),h,h).
48 bzd(c3(h,h,h),f,h,ph,h,h,n2(h,h),h,h).
49 bzd(c3(h,h,h),f,h,ph,h,h,n2(h,c2(o,n2(h,c3(h,h,h)))),h,h).
50 bzd(c3(h,h,c3(f,f,f)),h,h,ph,h,h,cl,h,h).
51 bzd(c3(h,h,c1(c1(h))),h,h,ph,h,h,cl,h,h).
52 bzd(c3(h,h,c2(h,c2(h,c3(h,h,h)))),h,h,ph,h,h,cl,h,h).
53 bzd(c3(c3(h,h,h),c3(h,h,h),c3(h,h,h)),h,h,ph,h,h,cl,h,h).
54 bzd(c3(h,h,c3(h,h,o1(c3(h,h,c2(o,n2(h,h)))))),
f,h,ph,h,h,cl,h,h).
55 bzd(c3(h,h,h),f,h,ph,cl,h,h,cl,h).
56 bzd(c3(h,h,h),h,h,ph,h,h,cl,cl,h).
57 bzd(h,h,h,ph,h,h,cl,h,cl).
58 bzd(h,h,h,ph,h,h,cl,h,c3(h,h,h)).
59 bzdtg(h,cl,h,ph,h,h,h,h,h).

60 bzdtg(h,h,h,ph,h,h,cl,h,h).
61 bzd(h,h,h,ph,c3(h,h,h),h,c3(h,h,h),h,h).
62 bzd(h,h,h,ph,cl,h,h,h,h).
63 bzd(c3(h,h,h),f,h,ph,cl,h,h,h,h).
64 bzd(c3(h,h,h),f,h,ph,h,h,h,h,cl).
65 bzd(h,h,h,cye,h,h,cl,h,h).
66 bzd(c3(h,h,h),h,h,cye,h,h,cl,h,h).
67 bzd(h,h,h,cya,h,h,cl,h,h).
68 bzd(h,h,h,naf,h,h,cl,h,h).
69 bzd(c3(h,h,h),h,c3(h,h,h),ph,h,h,cl,h,h).
70 bzd(h,h,o1(h),ph,h,h,cl,h,h).
71 bzd(c3(h,h,h),h,o1(h),ph,h,h,cl,h,h).
72 bzd(c3(h,h,h),f,cl,ph,h,h,cl,h,h).
73 bzd(c3(h,h,h),h,h,ph,h,h,c1(n),h,h).
74 bzd(c3(h,h,o1(c3(h,h,h))),h,h,ph,h,h,n2(o,o),h,h).
75 bzd(c3(h,h,c3(h,o1(h),c3(h,h,o1(h)))),f,h,ph,h,h,cl,h,h).
76 bzd(c3(c3(h,h,h),c3(h,h,h),c3(h,h,h)),
cl,h,ph,h,h,n2(o,o),h,h).
77 bzd(c3(h,h,h),h,o1(c2(o,n2(c3(h,h,h),c3(h,h,h)))),
ph,h,h,cl,h,h).

Table B.1. Training data for benzodiazepines: data set I.

#	Substituent	Target $\log 1/C$	Calcd Hansch	Error Hansch	Output CCS	Error CCS
1	R_7=CN, R_1=CH_3, $R_{2'}$=F	7,52	7,61	0,09	7,46	0,06
2	R_7=CH=CH_2, R_1=H, $R_{2'}$=H	7,62	7,66	0,04	7,28	0,34
3	R_7=H, R_1=H, $R_{2'}$=F	7,68	7,28	0,40	7,66	0,02
4	R_7=$COCH_3$, R_1=H, $R_{2'}$=F	7,74	7,80	0,06	7,91	0,17
5	R_7=CF_3, R_1=H, $R_{2'}$=H	7,89	7,74	0,15	7,90	0,01
6	R_7=Cl, R_1=CH_3, $R_{2'}$=H	8,09	7,78	0,31	7,99	0,10
7	R_7=Cl, R_1=CH_3, $R_{2'}$=Cl, $R_{6'}$=Cl	8,26	8,54	0,28	8,25	0,01
8	R_7=N_3, R_1=H, $R_{2'}$=F	8,27	8,01	0,26	8,27	0,00
9	R_7=NO_2, R_1=CH_3, $R_{2'}$=F	8,42	7,99	0,43	8,36	0,06
10	R_7=NO_2, R_1=H, $R_{2'}$=CF_3	8,45	8,90	0,45	8,43	0,02
11	R_7=I, R_1=CH_3, $R_{2'}$=F	8,54	8,39	0,15	8,54	0,00
12	R_7=Br, R_1=CH_3, $R_{2'}$=F, $R_{6'}$=F	8,62	8,44	0,18	8,60	0,02
13	R_7=Cl, R_1=H, $R_{2'}$=F	8,70	8,27	0,43	8,65	0,05
14	R_7=Cl, R_1=H, $R_{2'}$=Cl	8,74	8,60	0,14	8,69	0,05
15	R_7=NO_2, R_1=H, $R_{2'}$=Cl	8,74	8,70	0,04	8,76	0,02
16	R_7=NO_2, R_1=H, $R_{2'}$=F	8,82	8,15	0,67	8,66	0,16
17	R_7=F, R_1=CH_3, $R_{2'}$=F	8,29	7,82	0,47	8,26	0,03
18	R_7=F, R_1=CH_3, $R_{2'}$=H	7,77	7,48	0,29	7,72	0,05
19	R_7=F, R_1=H, $R_{2'}$=F	8,13	7,81	0,32	8,22	0,09
20	R_7=Cl, R_1=H, $R_{2'}$=H	8,03	7,80	0,23	7,86	0,17
21	R_7=Cl, R_1=H, $R_{2'}$=F, $R_{6'}$=F	8,79	8,33	0,46	8,68	0,11
22	R_7=Cl, R_1=CH_3, $R_{2'}$=F, $R_{6'}$=F	8,39	8,32	0,07	8,41	0,02
23	R_7=Cl, R_1=H, $R_{2'}$=Cl, $R_{6'}$=F	8,52	8,71	0,19	8,69	0,17
24	R_7=Cl, R_1=H, $R_{2'}$=Cl, $R_{6'}$=Cl	8,15	8,56	0,41	8,13	0,02
25	R_7=NO_2, R_1=H, $R_{2'}$=H	7,99	7,85	0,14	7,96	0,03
26	R_7=NO_2, R_1=CH_3, $R_{2'}$=Cl	8,66	8,63	0,03	8,79	0,13
27	R_7=Cl, R_1=CH_2CH_2OH, $R_{2'}$=F	7,61	8,31	0,70	7,48	0,13
28	R_7=Cl, R_1=H, R_3=(s)CH_3, $R_{2'}$=F	8,46	7,95	0,51	8,49	0,03
29	R_7=NO_2, R_1=H, R_3=(s)CH_3, $R_{2'}$=Cl	8,92	8,58	0,34	8,88	0,04
30	R_7=NO_2, R_1=CH_3, R_3=(s)CH_3, $R_{2'}$=F	8,15	8,22	0,07	8,13	0,02

(*Continued on next page*.)

Table B.1. (*continued*).

#	Substituent	Target $\log 1/C$	Calcd Hansch	Error Hansch	Output CCS	Error CCS
31	R_7=Br, R_1=H, R_5=2-pyridyl, $R_{2'}$=H	7,74	7,74	0,00	7,74	0,00
32	R_7=Cl, R_1=H, $R_{2'}$=Cl	8,03	8,76	0,73	8,08	0,05
33	R_7=H, R_1=H, $R_{2'}$=F, $R_{6'}$=F	7,72	7,12	0,60	7,78	0,06
34	R_7=H, R_1=CH_3, $R_{2'}$=F	7,85	7,40	0,45	7,86	0,01
35	R_7=H, R_1=CH_3, $R_{2'}$=Cl	8,42	7,94	0,48	8,27	0,15
36	R_7=H, R_1=H, $R_{2'}$=F, $R_{6'}$=F, R_8=Cl	7,55	7,46	0,09	7,46	0,09
37	R_7=H, R_1=H, $R_{2'}$=F, R_8=CH_3	7,72	7,47	0,25	7,70	0,02
38	R_7=Cl, R_1=H, $R_{2'}$=F, R_8=Cl	8,44	7,85	0,59	8,49	0,05
39	R_7=CH_3, R_1=H, $R_{2'}$=F, R_8=Cl	7,85	7,66	0,19	7,86	0,01
40	R_7=NH_2, R_1=CH_3, $R_{2'}$=H	6,34	6,65	0,31	6,32	0,02
41	R_7=NH_2, R_1=H, $R_{2'}$=H	6,41	6,32	0,09	6,59	0,18
42	R_7=H, R_1=H, $R_{2'}$=H	6,45	7,03	0,58	6,70	0,25
43	R_7=NHOH, R_1=CH_3, $R_{2'}$=F	7,02	6,66	0,36	6,98	0,04
44	R_7=NH_2, R_1=H, $R_{2'}$=Cl	7,12	7,24	0,12	7,18	0,06
45	R_7=CHO, R_1=H, $R_{2'}$=H	7,37	7,58	0,21	7,59	0,22
46	R_7=F, R_1=H, $R_{2'}$=H	7,40	7,48	0,08	7,43	0,03
47	R_7=C_2H_3, R_1=H, $R_{2'}$=H	7,44	7,39	0,05	7,39	0,05
48	R_7=NH_2, R_1=CH_3, $R_{2'}$=F	7,19	6,78	0,41	6,92	0,27
49	R_7=$NHCONHCH_3$, R_1=CH_3, $R_{2'}$=F	6,34	7,01	0,67	6,61	0,27
50	R_7=Cl, R_1=CH_2CF_3, $R_{2'}$=H	7,04	6,66	0,38	7,02	0,02
51	R_7=Cl, R_1=$CH_2C{\equiv}CH$, $R_{2'}$=H	7,03	7,72	0,69	7,10	0,07
52	R_7=Cl, R_1=$CH_2C_3H_5$, $R_{2'}$=H	6,96	6,97	0,01	7,02	0,06
53	R_7=Cl, R_1=$C(CH_3)_3$, $R_{2'}$=H	6,21	6,71	0,50	6,19	0,02
54	R_7=Cl, R_1=$(CH_2)_2OCH_2CONH_2$, $R_{2'}$=F	7,37	7,97	0,60	7,47	0,10
55	R_7=H, R_1=CH_3, $R_{2'}$=F, R_6=Cl, R_8=Cl	6,52	6,90	0,38	6,51	0,01
56	R_7=Cl, R_1=CH_3, $R_{2'}$=H, R_8=Cl	7,40	7,32	0,08	7,46	0,06
57	R_7=Cl, R_1=H, $R_{2'}$=H, R_9=Cl	7,43	7,19	0,24	7,37	0,06
58	R_7=Cl, R_1=H, $R_{2'}$=H, R_9=CH_3	7,28	7,43	0,15	7,27	0,01
59	R_7=H, R_1=H, $R_{2'}$=Cl	7,43	7,76	0,33	7,32	0,11
60	R_7=Cl, R_1=H, $R_{2'}$=H	7,15	7,93	0,78	7,21	0,06
61	R_7=CH_3, R_1=H, $R_{2'}$=H, R_6=CH_3	6,77	7,45	0,68	6,77	0,01
62	R_7=H, R_1=H, $R_{2'}$=H, R_6=Cl	6,49	6,93	0,44	6,34	0,15
63	R_7=H, R_1=CH_3, $R_{2'}$=F, R_6=Cl	6,82	7,39	0,57	6,99	0,17
64	R_7=H, R_1=CH_3, $R_{2'}$=F, R_9=Cl	7,14	7,39	0,25	7,21	0,07
65	R_7=Cl, R_1=H, R_5=cyclohexenyl, $R_{2'}$=H	7,47	7,34	0,13	7,45	0,02
66	R_7=Cl, R_1=CH_3, R_5=cyclohexenyl, $R_{2'}$=H	7,47	7,31	0,16	7,57	0,10
67	R_7=Cl, R_1=H, R_5=cyclohexyl, $R_{2'}$=H	7,06	7,21	0,15	7,05	0,01
68	R_7=Cl, R_1=H, R_5=naphthyl, $R_{2'}$=H	6,54	6,80	0,26	6,50	0,04
69	R_7=Cl, R_1=CH_3, R_3=(rac)CH_3, $R_{2'}$=H	7,31	7,39	0,08	7,34	0,03
70	R_7=Cl, R_1=H, R_3=(rac)OH, $R_{2'}$=H	7,74	7,95	0,21	7,85	0,11
71	R_7=Cl, R_1=CH_3, R_3=(rac)OH,$R_{2'}$=H	7,79	7,97	0,18	7,67	0,12
72	R_7=Cl, R_1=CH_3, R_3=(rac)Cl, $R_{2'}$=F	8,27	7,83	0,44	8,28	0,01

Table B.2. Test data for Benzodiazepines: data set I.

#	Substituent	Target $\log 1/C$	Calcd Hansch	Error Hansch	Ouput CCS	Error CCS
73	R_7=CN, R_1=CH_3, $R_{2'}$=H	6,42	7,43	1,01	6,84	0,42
74	R_7=NO_2, R_1=CH_2OCH_3, $R_{2'}$=H	6,37	7,71	1,34	6,90	0,53
75	R_7=Cl, R_1=$CH_2CHOHCH_2OH$, $R_{2'}$=F	6,85	7,74	0,89	7,53	0,68
76	R_7=NO_2, R_1=$C(CH_3)_3$, $R_{2'}$=Cl	6,52	8,27	1,75	6,99	0,47
77	R_7=Cl, R_1=CH_3, R_3=(rac)$OCON(CH_3)_2$, $R_{2'}$=H	6,05	7,42	1,37	7,56	1,51

The remaining Data Sets for Benzodiazepines are obtained by removing the racemic compounds (69,70,71,72,77), and using the following Test Sets:

Data Set II: 73,74,75,76;
Data Set III: 9,15,20,34,42;
Data Set IV: 15,23,52,68.

Notes

1. Backpropagation is only one particular way to implement the concept underlying the RAAM model.
2. Hydrogens atoms are excluded.
3. Since the maximization of the correlation is obtained using a gradient ascent technique on a surface with several maxima, a pool of hidden units is trained and the best one selected.
4. In order to characterize the fixed response, the drug concentration able to give half of the maximum response (IC_{50}) is commonly used.
5. Preliminary results on the application of our approach to this group of compounds were reported in [36].
6. No explicit representation of the atoms and bond type is required.
7. The root of a tree representing a benzodiazepine is determined by the common template, while the root for Alkanes is determined by the I.U.P.A.C. nomenclature system.
8. The multiplicity of the bond is implicitly encoded in the structure of the subtree.

References

1. D.S. Touretzky, "Boltzcons: Dynamic symbol structures in a connectionist network," *Artificial Intellicence*, vol. 46, pp. 5–46, 1990.
2. R. Rosenfeld and D.S. Touretzky, "Four capacity models for coarse-coded symbol memories," Technical Report CMU-CS-87-182, Carnegie Mellon, 1987.
3. J.B. Pollack, "Recursive distributed representations," *Artificial Intelligence*, vol. 46, no. 1/2, pp. 77–106, 1990.
4. G.E. Hinton, "Mapping part-whole hierarchies into connectionist network," *Artificial Intelligence*, vol. 46, pp. 47–75, 1990.
5. T.A. Plate, "Holographic reduced representations," *IEEE Transactions on Neural Networks*, vol. 6, no. 3, pp. 623–641, 1995.
6. P. Smolensky, "Tensor product variable binding and the representation of symbolic structures in connectionist systems," *Artificial Intelligence*, vol. 46, pp. 159–216, 1990.
7. A. Sperduti and A. Starita, "Supervised neural networks for the classification of structures," *IEEE Transactions on Neural Networks*, vol. 8, no. 3, pp. 714–735, 1997.
8. P. Frasconi, M. Gori, and A. Sperduti, "A framework for adaptive data structures processing," *IEEE Transactions on Neural Networks*, vol. 9, no. 5, pp. 768–786, 1998.
9. C. Hansch, P.P. Maloney, T. Fujita, and R.M. Muir, *Nature*, vol. 194, pp. 178–180, 1962.
10. C. Hansch and T. Fujita, *J. Am. Chem. Soc.*, vol. 86, pp. 1616–1626, 1964.
11. Dimitra Hadjipavlou-Litina and Corwin Hansch, "Quantitative structure-activity relationships of the benzodiazepines. A review and reevaluation, *Chemical Reviews*, vol. 94, no. 6, pp. 1483–1505, 1994.
12. L.H. Hall and L.B. Kier, "The molecular connectivity chi indexes and kappa shape indexes in structure-property modeling," in *Reviews in Computational Chemistry*, VCH Publishers, Inc., New York, ch. 9, pp. 367–422. 1991.
13. J. Zupan and J. Gasteiger, *Neural Networks for Chemists: An introduction*, VCH Publishers: NY(USA), 1993.
14. Ajay, "A unified framework for using neural networks to build QSARs," *J. Med. Chem.*, vol. 36, pp. 3565–3571, 1993.
15. D. Cherqaoui and D. Villemin, "Use of neural network to determine the boiling point of alkanes," *J. Chem. Soc. Faraday Trans.*, vol. 90, no. 1, pp. 97–102, 1994.
16. C. Goller, "A Connectionist Approach for Learning Search-Control Heuristics for Automated Deduction Systems," Ph.D. Thesis, Technical University Munich, Computer Science, 1997.
17. A. Sperduti, D. Majidi, and A. Starita, "Extended cascade-correlation for syntactic and structural pattern recognition," in *Advances in Structural and Syntactical Pattern Recognition*, edited by P. Perner, P. Wang, and A. Rosenfeld, Springer-Verlag, Berlin, pp. 90–99, 1996. *Lecture notes in Computer Science*, vol. 1121.
18. H. Kubinyi, *Burger's Medicinal Chemistry and Drug Discovery*, vol. 1, fifth edn., John Wiley & Sons, Inc: New York, pp. 528–530, 1995.
19. H. Kubinyi and U. Abraham, "Practical problems in PLS analyses," in *3D-QSAR in Drug Design. Theory Methods and Application*, edited by H. Kubinyi, ESCOM: Leiden, 1993, pp. 717–728.
20. D. Rogers and A.J. Hopfinger, "Application of genetic function approximation to quantitative structure-activity relationships and quantitative-property relationships," *J. Chem. Inf. Comput. Sci.*, vol. 34, no. 4, pp. 854–866, 1993.
21. S.S. So and M. Karplus, "Evolutionary optimization in quantitative structure-activity relationship: An application of genetic

neural networks," *J. Med. Chem.*, vol. 39, pp. 1521–1530, 1996.
22. D.H. Rouvray, "Should we have designs on topological indices ?," in *Chemical Applications of Topology and Graph Theory*, edited by R.B. King, Elsevier Science Publishing Company, pp. 159–177, 1983.
23. J.A. Burns and G.M. Whitesides, "Feed-forward neural networks in chemistry: Mathematical system for classification and pattern recognition," *Chemical Reviews*, vol. 93, no. 8, pp. 2583–2601, 1993.
24. D.H. Rouvray, *Computational Chemical Graph Theory*, Nova Science Publishers: New York, p. 9, 1990.
25. M. Barysz, G. Jashari, R.S. Lall, V.K. Srivastava, and N. Trinajstic, "On the distance matrix of molecules containing heteroatoms," in *Chemical Applications of Topology and Graph Theory*, edited by R.B. King, Elsevier Science Publishing Company, pp. 222–230, 1983.
26. V.R. Magnuson, D.K. Harris, and S.C. Basak, "Topological indices based on neighborhood symmetry: Chemical and biological application," in *Chemical Applications of Topology and Graph Theory*, edited by R.B. King, Elsevier Science Publishing Company, pp. 178–191, 1983.
27. Y. Suzuki, T. Aoyama, and H. Ichikawa, "Neural networks applied to quantitative structure-activity relationships," *J. Med. Chem.*, vol. 33, pp. 2583–2590, 1990.
28. A.F. Duprat, T. Huynh, and G. Dreyfus, "Towards a principled methodology for neural network design and performance evaluation in QSAR; Application to the prediction of LogP," *J. Chem. Inf. Comput. Sci.*, vol. 38, no. 4, pp. 586–594, 1999.
29. L.K. Peterson, "Quantitative structure-activity relationships in carboquinones and benzodiazepines using counter-propagation neural networks," *J. Chem. Inf. Comput. Sci.*, vol. 35, no. 5, pp. 896–904, 1995.
30. Shuhui Liu, Ruisheng Zhang, Mancang Liu, and Zhide Hu, "Neural networks-topological indices approach to the prediction of properties of alkene," *J. Chem. Inf. Comput. Sci.*, vol. 37, pp. 1146–1151, 1997.
31. D.W. Elrod, G.M. Maggiora, and R.G. Trenary, "Application of neural networks in chemistry, 1. Prediction of electrophilic aromatic substitution reactions," *J. Chem. Inf. Comput. Sci.*, vol. 30, pp. 447–484, 1990.
32. V. Kvasnička and J. Pospichal, "Application of neural networks in chemistry.prediction of product distribution of nitration in a series of monosubstituted benzenes," *J. Mol. Struct. (Theochem)*, vol. 235, pp. 227–242, 1991.
33. S.E. Fahlman and C. Lebiere, "The cascade-correlation learning architecture," in *Advances in Neural Information Processing Systems 2*, edited by D.S. Touretzky, San Mateo, CA: Morgan Kaufmann, pp. 524–532, 1990.
34. S.E. Fahlman, "The recurrent cascade-correlation architecture," Technical Report CMU-CS-91-100, Carnegie Mellon, 1991.
35. S. Muggleton and L. De Raedt, "Inductive logic programming: Theory and methods," *Journal of Logic Programming*, vol. 19, 20, pp. 629–679, 1994.
36. A.M. Bianucci, A. Micheli, A. Sperduti, and A. Starita, "Quantitative structure-activity relationships of benzodiazepines by Recursive Cascade Correlation," in *Proceedings of IJCNN '98 - IEEE World Congress on Computational Intelligence*, Anchorage, Alaska, May 1998, pp. 117–122.

Anna Maria Bianucci received the "laurea" degree in Chemistry (1974) from the University of Pisa, Italy, and the Specialization degree in Automatic Calculus (1976) from the Information Processing Institute of Pisa. She was "Research and Teaching Fellow" of the Italian Ministry of Education at the Institute of Pharmaceutical Chemistry of Pisa and then she became Researcher for the same institution (1981). In 1986 and 1987 she was at the Department of Pharmaceutical Chemistry, University of California at San Francisco, where she did research in the NMR laboratory. She is now Researcher at the Department of Pharmaceutical Sciences of the University of Pisa. Her scientific interests have been focused in designing new biologically active compounds through direct study of drug-receptor interactions detailed at molecular level, and more recently through QSAR methodologies. She has been responsible for several research projects.

Alessio Micheli received the "laurea" degree in Computer Science (1998) from the University of Pisa, Italy. He is currently collaborating with the Computer Science Department of the University of Pisa. His research interests are mainly in neural networks for the processing of structured information and QSPR/QSAR.

Alessandro Sperduti received his university education from the University of Pisa, Italy ("laurea" and Doctoral degrees in 1988 and 1993, respectively, all in Computer Science.) In 1993 he spent a period at the International Computer Science Institute, Berkeley, supported by a postdoctoral fellowship. In 1994 he moved back to the Computer Science Department, University of Pisa, where he was Assistant Professor, and where he presently is Associate Professor. His research interests include pattern recognition, image processing, neural networks, hybrid systems. In the field of hybrid systems his work has focused on the integration of symbolic and connectionist systems for the processing of structured information. He contributed to the organization of several workshops on this subject and he served also in the program committee of conferences on Neural Networks. Alessandro Sperduti is the author of around 60 refereed papers mainly in the areas of Neural Networks, Fuzzy Systems, Pattern Recognition, and Image Processing.

Antonina Starita graduated in Physics at the University of Naples and postgraduated in Computer Science at the University of Pisa. She was research fellow of the Italian National Council of Research at the Information Processing Institute of Pisa and then she became researcher for the same institution. Since 1980, she has become Associate Professor at the Computer Science Department of the University of Pisa, where she is in charge of the two regular courses "Knowledge acquisition and Expert Systems" and "Neural Networks" for the Degree in Computer Science of the Science Faculty of the University of Pisa. She has been active in the areas of Biomedical signal processing, Rehabilitation Engineering and Biomedical applications of signal processing, being responsible of many research projects at national and international level. Her current scientific interests are now shifted to AI methodologies, Robotics, Neural Networks, Hybrid Systems, Sensory Integration in Artificial Systems. She is member of the Italian Medical and Biological Engineering Society, of the International Medical and Biological Engineering Society, of the IEEE (Institute of Electrical and Electronics Engineers) and of the INNS (International Neural Network Society).