Adaptive Structure Processing with ANN: Is it useful for Chemical Applications?

Christoph Goller
Institut für Informatik, Technische Universität München
Arcisstr. 21, D-80290 Muenchen, Germany
Email: goller@informatik.tu-muenchen.de
WWW: http://wwwjessen.informatik.tu-muenchen.de/personen/goller.html

Abstract

During the last years, new connectionist approaches for adaptive structure processing called folding architecture networks or recursive neural networks have been developed. The main objective of this paper is to explore the applicability of these networks in the field of chemistry. Experimental results on a benchmark for the prediction of quantitative structure activity relationships are presented and compared to results achieved with other machine learning techniques. Though the results achieved with the new networks are slightly better, it has to be stated that the statistical evidence is rather weak. For future comparisons, bigger benchmarks are needed.

Introduction

One of the fundamental problems of methods from the fields of artificial neural networks and from statistical pattern recognition is that they cannot deal with structured objects. With *structured objects* we mean objects that are composed of 'smaller' objects, which may be structured too. Examples for such objects are chemical structures, algebraic (mathematical) expressions and formulas, software source code, and conceptual and taxonomic graphs. Though the objects that are usually considered are finite, the size of objects within one domain is often not limited and one normally has objects of very different size within one domain. This contrasts with the *static* kind of data (*fixed-length real vectors*) usually handled by statistical pattern recognition or neural network approaches.

During the last years, new methods for *adaptive structure processing*, viz. *folding architecture networks* (*FAs*) (Goller & Küchler 1996; Goller 1997) and the closely related concept of *recursive neural networks* (*RNs*) (Frasconi, Gori, & Sperduti 1998), have been developed. These networks are able to solve supervised learning tasks (such as classification and prediction) for structured objects. Applications of *FAs* and *RNs* in the fields of automated deduction, and logo recognition are described e.g. in (Schulz, Küchler, & Goller 1997; Goller 1997; Frasconi *et al.* 1997).

In this paper we give a brief introduction to *FAs* and we argue that their ability to process structured objects strongly suggests their application in chemistry. However, in order to apply *FAs* one has to represent chemical structures as *labelled trees*. Since chemical structures are usually represented as *cyclic undirected graphs* this is a non-trivial problem for which we don't have a general solution. We present a special solution in case of a standard benchmark for *quantitative structure activity relationship* (*QSAR*) prediction (Slipo & Hansch 1975; Hirst, King, & Sternberg 1994) for which 2-D representations of chemical structures are transformed into labelled trees. Furthermore, experimental results on this benchmark are presented and compared with results achieved with linear regression, standard feed-forward networks and inductive logic programming (Hirst, King, & Sternberg 1994).

Folding Architecture Networks

FAs (Goller & Küchler 1996; Goller 1997) have been developed to solve supervised learning tasks involving structured objects. The structured objects that can be handled by *FAs* are *finite labelled ordered trees* in the following simply called labelled trees. The labels are assumed to be real vectors of fixed uniform length. To each node a label is attached and there are no labels for edges. The order concerns the children of a node and we assume that there is a limit (*maximum out-degree*) for the number of children. Furthermore, each labelled tree has exactly one *root-node*. Logical terms or algebraic expressions over a finite set of symbols (signature) can e.g. easily be represented as labelled trees by choosing the maximum arity of all symbols as maximum out-degree and introducing a function that maps the symbols to real vectors.

FAs are a recursive variant of the standard multilayer feed-forward network and can be seen as a generalization of recurrent neural networks. They can be used to learn approximations to mappings from labelled trees to $\mathbb{R}^n$ based on *samples* for these mappings. A *FA* is composed of two multi-layer feed-forward networks, viz. the recursively applied *encoder network* N_{enc} and the *transformation network* N_{trans}. Usually sigmoid activation functions are considered. N_{enc} is used to compute a *compressed representation* (a fixed length

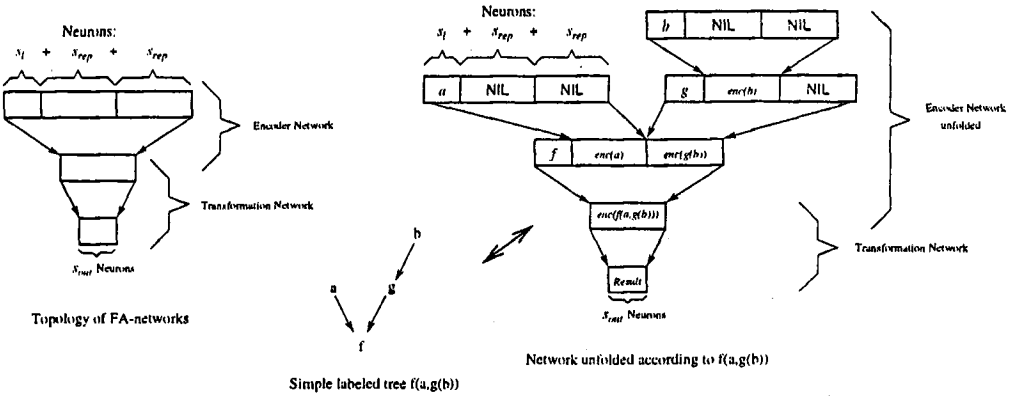


Figure 1: Example of a *FA* processing the logical term $f(a, g(b))$.

vector) for a given labelled tree while N_{trans} takes this representation as input and computes the final output of the whole network. Fig. 1 shows an example of a *FA* which can process labelled trees with a maximum out-degree of two. In this example both N_{enc} and N_{trans} are single-layer feed-forward networks. Note that in case of a maximum out-degree of one and single-layer N_{enc} and N_{trans} we get a simple recurrent network.

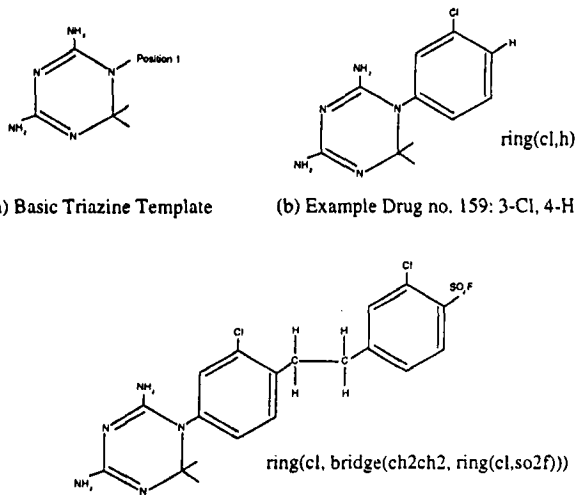
Let *Tree* represent the domain of labelled trees that have to be processed with a maximum out-degree of $out_{max} \in \mathbb{N}$. Furthermore, let $s_l, s_{rep}, s_{out} \in \mathbb{N}$ denote the dimensions of labels, compressed representations, and the network's final output, respectively. Then N_{enc} has $s_{tot} = s_l + out_{max} \times s_{rep}$ input neurons and s_{rep} output neurons. Thus, N_{enc} computes a function $f_{enc} : \mathbb{R}^{s_l + out_{max} \times s_{rep}} \rightarrow \mathbb{R}^{s_{rep}}$. Assume we have a labelled tree $t = l(t_1, \ldots, t_n)$, where $l \in \mathbb{R}^{s_l}$ is the label of the root-node of t and $t_1, \ldots, t_n \in Tree$ is the ordered list of children of t. Let $\bullet$ denote vector concatenation. Then the function $enc : Tree \rightarrow \mathbb{R}^{s_{rep}}$ which computes the compressed representations is defined by $l(t_1, \ldots, t_n) := f_{enc}(l \bullet enc(t_1) \bullet \ldots \bullet enc(t_n) \bullet \mathrm{NIL}^{(a_{max}-n)})$. We say the network is *virtually unfolded* to encode t. Leaf-nodes are encoded first and composite trees are encoded using the representations of their children. $\mathrm{NIL} \in \mathbb{R}^{s_{rep}}$ is a fixed representation indicating the absence of a child. The transformation network N_{trans} has s_{rep} input neurons and s_{out} output neurons. Thus, it computes a function $f_{trans} : \mathbb{R}^{s_{rep}} \rightarrow \mathbb{R}^{s_{out}}$. By using the output of the encoder network N_{enc} as the input for N_{trans}, the total network computes a function $f_{trans} \circ enc : Tree \rightarrow \mathbb{R}^{s_{out}}$. See also Fig. 1.

It has been shown that *FAs* are universal approximators for functions from labelled trees to real vector spaces (Hammer & Sperschneider 1997). Furthermore, any bottom-up tree-automaton can be simulated by a *FA* (Küchler 1998).

FAs can be trained by *back-propagation through structure* (*BPTS*), a recursive variant of standard back-propagation. *BPTS* means that the error at the output layer of N_{trans} is propagated back through the entire unfolded network. In this way the exact gradient with respect to the weights in N_{trans} and N_{enc} is computed and the training is completely supervised. Current refinements include an efficient representation of all labelled trees of a training set as one minimal directed acyclic graph (*DAG*) by representing identical subtrees which occur at different positions only once. Since the time complexity of gradient computation with *BPTS* is linear to the number of nodes and edges in the training set, the minimal *DAG*-representation can speed up the gradient computation considerably.

The central problem for using statistical pattern recognition or neural approaches in domains of structured objects is that these approaches can only handle static data. Thus, fixed-length vector representations of the structured objects are needed. One way to get such representations are *features-vectors*. However, the definition and selection of features already introduces a very strong *bias* and severely limits the class of functions and relations that can be expressed or learned for the respective domain of structured objects. Another possibility works by a priori reserving for each possible possition in a structured object a region of the fixed-length representation. However, this means that the size of the structured objects that can be treated has to be limited a priori. Furthermore, these representations are very big for non-trivial domains of structures and they usually contain many void regions caused by the varying size and structure of the objects. Note that this approach for getting fixed-length vector representations for structures was also applied in the linear regression- and standard feed-forward network experiments on *QSAR*-prediction from (Hirst, King, & Sternberg 1994) which serve as reference point for our experimental results.

In case of *FAs* the size of the labelled trees that can be handled is not limited. Only the maximum number of children of a node has to be limited a priori. Furthermore, no definition of features of the structured objects



(a) Basic Triazine Template (b) Example Drug no. 159: 3-Cl, 4-H

(c) Example Drug no. 173: 3-Cl, 44-$(CH_2)_2C_6H_3$-3'-Cl, 4'-SO_2F

Figure 2: Triazine template and example drugs.

is needed. Instead, the compressed representations that are generated for the labelled trees by a trained *FA* are exclusively optimized for the respective supervised learning task. It can be expected that they contain exactly the kind of information necessary for computing the desired output. These advantages strongly suggest the application of *FAs* in the field of chemistry.

Representing Chemical Structures as Labelled Trees

For our experiments a standard benchmark for *QSAR*-prediction was chosen (Slipo & Hansch 1975; Hirst, King, & Sternberg 1994). It deals with the inhibition of E. coli dihydrofolate reductase by triazines. Triazines act as anti-cancer agents by inhibiting the enzyme dihydrofolate reductase. In this way they inhibit the reproduction of cancer cells. In total, the benchmark set consists of 186 different drugs and their respective activities. The triazines of the benchmark are composed of 87 basic chemical groups, i.e. atoms or small sets of structurally connected atoms. Each group is described by discrete physicochemical attributes, i.e. polarity, polarisability, H-bond donor, H-bond acceptor, π-donor, π-acceptor, size, flexibility, σ-effect, and branching. These attributes are part of the benchmark and have been determined by experts.

The 2-D representations of triazines (Fig. 2) are cyclic undirected graphs. In order to apply *FAs*, a suitable representation for triazines as logical terms or labelled trees is needed. Fortunately, all triazines in the benchmark set have a common template (Fig. 2a). This template is not explicitly represented. Instead it is used to define the root node of the tree-representation of a triazine. For most of the triazines, there is a phenyl ring at position 1 of the basic template (Fig. 2b). Phenyl rings have varying substituents at positions 3 and 4. For the tree-representation, the cycle of the phenyl ring is simply represented as one node with the label `ring` leading to `ring`(R_3, R_4). The variables R_3 and R_4 represent the tree-representations for possible substituents at the respective positions. In order to keep the tree-representation as simple as possible, the hydrogen atoms at positions 2, 5 and 6 of phenyl rings are not explicitly represented. Each substituent of a phenyl ring at positions 3 and 4 can consist of a basic group or a ring structure. In the second case it consist of a bridge and a further phenyl ring (Fig. 2c). For the tree-representation, the symbol `bridge` is introduced. It defines a bridge to a second phenyl ring. The first argument describes the chemical structure of the bridge and the second one the further phenyl ring and its substructures.

Of course several other tree-representation for triazines (e.g. more detailed representations) are possible. The benefit of the chosen tree-representation is, that it allows also an easy integration of the physicochemical attributes. A constant (leaf) which represents a chemical group can be extended to a function symbol (inner node of a tree). The arguments are then used to introduce the physicochemical attributes of the chemical groups to the tree- representation. For example the chlorine atom's attribute tree-representation is: `cl(po(polar3, polarisable1), hy(h_donor0, h_acceptor0), pi(pi_donor1, pi_acceptor0), size1, flex0, sigma3, branch0)`. Note that related properties are encapsulated by special function symbols (`po`, `hy`, `pi`). This reduces the maximum arity of all employed operators and keeps the *FA* topology as small as possible. For a more detailed discussion of the tree-representations see (Schmitt 1997).

Experimental Results

For the experiments, six-fold cross-validation was used and the folds that were used are identical to those used in (Hirst, King, & Sternberg 1994). This allows a direct comparison of the results. Tab. 1 summarizes the parameters and the results of the different experiments performed. Column two displays the machine learning technique used. The results for linear regression, standard feed-forward networks and *Golem* are taken from (Hirst, King, & Sternberg 1994). For *FAs*, the employed topology is displayed. The next column describes the "knowledge" used for training. For linear regression and feed-forward networks the already described fixed-length vector representation with one region for every possible position was used. Accuracy of the different approaches is measured by the Spearman's rank correlation coefficient between the actual and predicted activities on the training and testing sets. The means ($\overline{r_{xy}}$) and standard deviations ($s_{r_{xy}}$) were computed from six-fold cross-validation trials and are displayed in columns four and five. For the *FA* experiments, two different kinds of training and testing sets were used. The first kind contained only tree-

No.	Machine learning technique $s_{tot}(s_l + out_{max} * s_{rep})//s_{rep}//s_{out}$	"Knowledge"	Training set $\overline{r_{xy}}$	Training set $s_{r_{xy}}$	Testing set $\overline{r_{xy}}$	Testing set $s_{r_{xy}}$
1.	Linear regression	60 real valued attrib. + squares	0.540	0.046	0.446	0.181
2.	Feed-forward network	60 real valued attributes	0.799	0.099	0.481	0.145
3.	*Golem* (inductive logic prog.)	see experiments 7 - 9	0.633	0.027	0.431	0.166
4.	*FA*: $14(8+2*3)//3//1$	simple tree-representation	0.614	0.096	0.435	0.092
5.	*FA*: $18(8+2*5)//5//1$	simple tree-representation	0.701	0.100	0.367	0.177
6.	*FA*: $22(8+2*7)//3//1$	simple tree-representation	0.720	0.119	0.435	0.131
7.	*FA*: $29(8+7*3)//3//1$	tree(term)-representation + physicochemical attributes	0.656	0.094	0.393	0.098
8.	*FA*: $43(8+7*5)//5//1$	tree(term)-representation + physicochemical attributes	0.723	0.099	0.477	0.152
9.	*FA*: $57(8+7*7)//7//1$	tree(term)-representation + physicochemical attributes	0.750	0.124	0.526	0.044

Table 1: Experimental results on the triazine benchmark.

representations of the triazines, i.e. all chemical groups were represented as constants (rows 4 to 6). The second kind consisted of tree-representations with physicochemical attributes. A binary encoding scheme (using -1 and $+1$) was used to encode the node-labels (the symbols `ring` and `bridge`, the basic chemical groups, and the physicochemical attributes) and uniqueness of the labels was guaranteed. The activities of the triazines were rescaled to the interval $[-1, +1]$. In contrast to the experiments presented in (Schmitt & Goller 1998), the testing sets were not used to decide when training of the *FAs* had to stop. The testing sets, which contained 31 triazines, were split into monitor sets with 30 elements and new testing sets with 1 element. The Spearman's ranks on the monitor sets were used to determine when training had to stop. Since we used every possible split, 31 different *FAs* were trained for every cross-validation trial of every experiment (row in Tab. 1). The final generalization was always measured on the one remaining triazine which was neither in the training nor in the monitor set.

Conclusion and Outlook

In contrast to preliminary results presented in (Schmitt & Goller 1998) we are not able to further support our claim of a clear superiority of our approach. Though the generalization performance in experiment 9 from Tab. 1 is better than in all other experiments, the difference is too small to give statistical evidence. It seems that the performances of all methods (including ours) applied so far to this benchmark are more or less comparable. A possible reason is that already a lot of effort has been put into the task of representing the chemical structures of the benchmark as fixed-length vectors (e.g. the definition of physicochemical attributes) suitable for standard neural networks and statistical methods. The structures of the benchmark are not very complex and by transforming them to labelled trees (terms) further complexity is lost. Note that we basically use the same transformation as the inductive logic programming approach. Thus the final structures used by our *FAs* and by the inductive logic programming approach are very small, possibly too small to really benefit from the special abilities to handle structured objects. Furthermore, the benchmark is quite small (186 compounds) at least for statistical or neural network learning approaches. For future work more general and more elaborate methods to represent chemical structures as labelled trees are needed. Methods for representing 3-D information as labelled trees are of particular interest, Furthermore, we are looking for new and bigger benchmarks to experiment with.

References

Frasconi, P.; Gori, M.; Marinai, S.; Sheng, J.; ; Soda, G.; and Sperduti, A. 1997. Logo Recognition by Recursive Neural Networks. In *Proceedings of GREC97*, 144–151.

Frasconi, P.; Gori, M.; and Sperduti, A. 1998. A general framework for adaptive processing of data structures. *IEEE Transactions on Neural Networks* 9(5).

Goller, C., and Küchler, A. 1996. Learning Task-Dependent Distributed Representations by Backpropagation Through Structure. In *Proceedings of the ICNN-96*, 347–352. IEEE.

Goller, C. 1997. *A Connectionist Approach for Learning Search-Control Heuristics for Automated Deduction Systems.* Ph.D. Dissertation, Technical University Munich, Computer Science.

Hammer, B., and Sperschneider, V. 1997. Neural Networks can approximate Mappings on Structured Objects. In P.Wang, P., ed., *Second International Conference on Computational Intelligence and Neuroscience*, 211–214.

Hirst, J.; King, R.; and Sternberg, M. 1994. Quantitative structure-activity relationships by neural networks and inductive logic programming. II. The inhibition of dihydrofolate reductase by triazines. *Computer-Aided Molecular Design* (8):421–432.

Küchler, A. 1998. On the Correspondence between Neural Folding Architectures and Tree Automata. Technical Report 98-06, Dept. of Neural Information Processing, Computer Science, University of Ulm.

Schmitt, T., and Goller, C. 1998. Relating Chemical Structure to Activity with the Structure Processing Neural Folding Architecture. In *Proceedings of the EANN'98.*

Schmitt, T. 1997. Evaluation of the Neural Folding Architecture for Inductive Learning Tasks concerning Logical Terms and Chemical Structures. Master's thesis. Institut für Informatik, Technische Universität München.

Schulz, S.; Küchler, A.; and Goller, C. 1997. Some Experiments on the Applicability of Folding Architecture Networks to Guide Theorem Proving. In *Proceedings of the Flairs-97.*

Slipo, C., and Hansch, C. 1975. *American Chemical Society* 97:6849.