ESANN'1999 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 21-23 April 1999, D-Facto public., ISBN 2-600049-9-X, pp. 27-32

Learning in Structured Domains

Marco Gori
Dipartimento di Ingegneria dell'Informazione
Università di Siena
Via Roma n. 56, Siena, Italy
E-mail: marco@ing.unisi.it

Abstract

By and large, learning from examples in the machine learning literature refers to static data types. That main stream of interest, however, has had significant bifurcations (see e.g. the learning issues connected with syntactic and structured pattern recognition) arisen from the need to exploit the structure inherently attached to the data of some learning tasks.

In this paper, I review briefly the research carried out in the last few years in the area of connectionist models in the attempt to extend the corresponding learning approaches to the case of structured domain. I give a unified picture of the adaptive computation which can be carried out on graphical objects and show that, under certain restrictions on the kind of graph to be processed, the classic learning algorithm for feedforward networks can be straightforwardly extended.

1 Introduction

Most interesting intelligent information processing tasks are based on the presence of some form of structure that, however, cannot easily be grasped at a symbolic level. The data associated with interesting real-world problems is often inherently structured. On the other hand, the truly sub-symbolic nature of many of those problems makes it very hard to extract clean structured symbolic data. The way humans process this kind of information can be regarded neither as strictly symbolic nor subsymbolic, neither sequential nor parallel, but is in fact a sort of magic mixture of all of these. Not only is the human brain a complex graph of elementary units, but data to be processed can often be regarded as complex structures of elementary units themselves.

Most symbolic and sub-symbolic models of learning, including neural networks, are conceived for processing static data types. The learning of sequential information is the first step toward the adaptive computation of dynamic data types. Recurrent neural networks were conceived so as to exhibit a dynamic behavior for incorporating time and, more recently, they have been extended to the processing of structured information. In this paper, I give a general

framework for adaptive computation of structured information [1] and focuses attention on connectionist models presenting either architecture or learning issues. This formulation extends recurrent neural networks significantly [2] and makes it possible to conceive graphical models with neuronal units which process data represented by graphs themselves.

2 Links with the related research

The need for structured representations in the field of pattern recognition was early stimulated by Norber Wiener who proposed the concept of pattern as an *arrangement characterized by the order of the elements of which it is made, rather than by the intrinsic nature of these elements.* The development of the theory of formal languages, which can be traced to the middle 1950s by Noam Chomsky, was the main catalyst of the research in the field of syntactical pattern recognition, where the role of the structure in the patterns was particularly stressed. Whereas the approaches to syntactic pattern recognition are well-suited for taking the structure of the pattern into account, most adaptive pattern recognition techniques seem to face effectively the sub-symbolic nature of patterns, but can hardly be regarded as appropriate models for processing structured information.

For sequence, or lists, special connectionist models referred to as recurrent neural networks have been proposed either for representational or learning issues. The interest in dynamical recurrent networks can be traced back to earliest work by McCulloch and Pitts. The authors had already incorporated the time dimension into neural networks that, however, were based on thresholding non-linearities. A nice interpretation of dynamic neural networks for processing temporal information is that of reducing the network to a static architecture by time-unfolding. The ideas of time-unfolding and Backpropagation through time has originated the concept of encoding network which will be reviewed in this paper. In the literature, the recurrent neural networks for processing sequences have been massively investigated in the last few years in a number of different fields, like speech recognition, natural language processing, time series forecasting, and automatic control (see e.g. [3, 4]).

The difficulties related to the representation of complex structures into connectionist models were early mentioned by Hinton in its preface to the "Special Issue on Connectionist Symbol Processing [5]. He pointed out that one of the basic problems is to devise effective ways of representing complex structures without sacrificing the ability to learn.

The recursive auto-associative memories (RAAM) are one step further the representation and the learning of sequences. They were proposed as a model for processing trees with labels attached only to the leaves [6]. In that paper, Pollack seems to address some of Fodor & Phylyshyn's [1] main arguments against connectionism, by pointing out that the RAAM is a connectionist ar-

[1] A more comprehensive treatment of the topic can be found in [2].
[2] In this paper, recurrent networks will be also referred to as recursive networks.

ESANN'1999 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 21-23 April 1999, D-Facto public., ISBN 2-600049-9-X, pp. 27-32

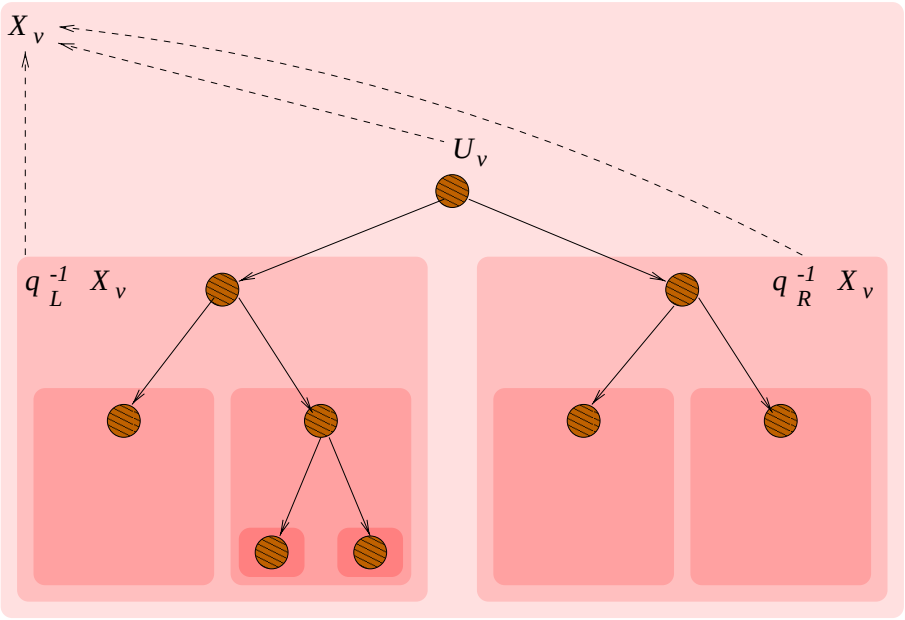

Figure 1: The generalization of recurrent neural networks to the processing of structured information. For instance, in the case of binary trees the state associated with a given node v depends on the attached label $\boldsymbol{U}_v$ and on the states $\boldsymbol{X}_{v,L}$ and $\boldsymbol{X}_{v,R}$ of its children.

chitecture which is capable of discovering compact representations for compositional symbolic structures. A remarkable extension of RAAM for considering labelled graphs was proposed in [7].

3 Recursive networks for processing DOAG's

Let us consider a directed ordered graph so as for any node v one can identify a set, eventually empty, of ordered children $ch[v]$. For each node we can consider the following state equations

$$\begin{aligned} \boldsymbol{X}_v &= f\left(\boldsymbol{X}_{ch[v]}, \boldsymbol{U}_v, v, \Theta_v^f\right) \\ \boldsymbol{Y}_v &= g\left(\boldsymbol{X}_v, v, \Theta_v^g\right). \end{aligned} \tag{1}$$

The rationale behind this model is well illustrated in Figure 1 for the case of binary trees. The state associated with each node is calculated as a function of the attached label and of the states associated with the left and right children, respectively. The operators q_L^{-1} and q_R^{-1} make it possible to address the information associated with the left and right children of a given node and, therefore, generalize straightforwardly the temporal delay operator q^{-1}. Of course, for any node, the children must be ordered so as to be able to produce different outputs for binary trees $\{r, L, R\}$ and $\{r, R, L\}$. Lists are just a special case of binary tree in which one of the children is null.

ESANN'1999 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 21-23 April 1999, D-Facto public., ISBN 2-600049-9-X, pp. 27-32

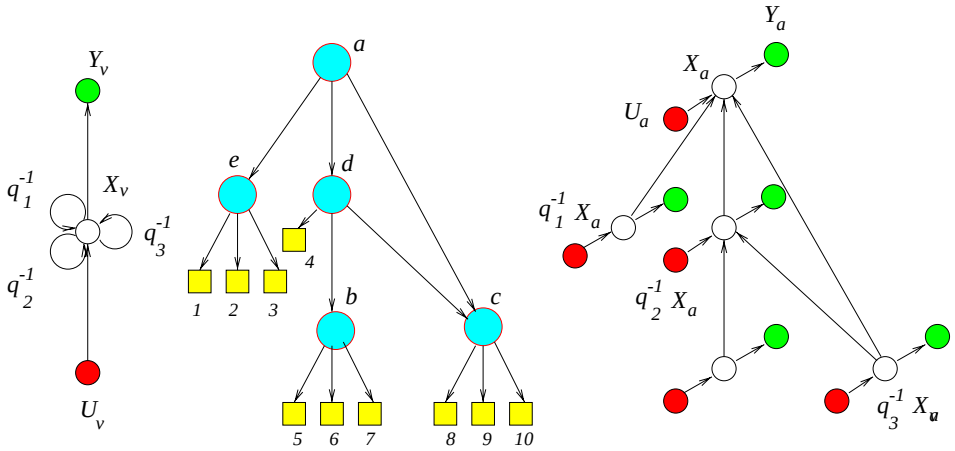

Figure 2: Construction of the encoding network corresponding to a recursive network and a DOAG. Note that $o = 3$ (graph outdegree), and that the nil pointers are represented by proper frontier (initial) states.

From the encoding network depicted in Figure 2 we can see a pictorial representation of the computation taking place in the recursive neural network. Each `nil` pointer is associated with a *frontier state* $\hat{\boldsymbol{X}}_v$, which is in fact an initial state that turns out to be useful to terminate the recursive equation. The graph plays its own role in the computation process either because of the information attached to its nodes or for its topology. A formal description of the computation of the input graph requires sorting the nodes, so as to define for which nodes the state can be computed first. In the literature, this problem is referred to as *topological sorting*.

4 Connectionist architectures for recursive networks

The graphical models presented in the previous section emphasize the structure of independence of some variables in the state-based model of equation 1. For instance, a classic structure of independence arises when the connections of any two state variables $\boldsymbol{X}_v$ and $\boldsymbol{X}_w$ only take place between components $X_{i,v}$ and $X_{w,i}$ with the same index i. In the case of lists, and consequently of sequences, this assumption means that only *local-feedback* connections are permitted for the state variables. The information attached to the recursive network, however, needs to be integrated with a specific choice of functions f and g which must be suitable for learning the parameters. Let o be the maximum outdegree of the given directed graph. The dependence of the node v on its children $ch[v]$ can be expressed by *pointer matrices* $\boldsymbol{A}_v(k) \in \mathcal{R}^{n,n}$, $k = 1, \ldots o$. Likewise, the information attached to the nodes can be propagated by weight matrix

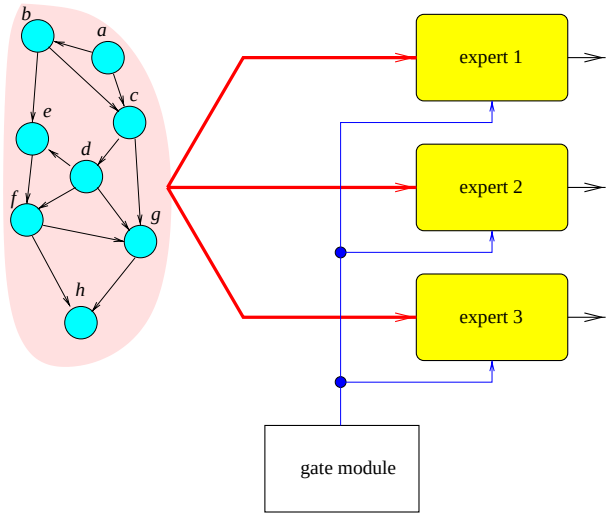

Figure 3: A typical example of non-stationarity: DOAG's with a remarkably different outdegree. For instance, one can divide the examples into three classes (small, medium, large outdegree) and use different recursive neural networks for each class.

$\boldsymbol{B}_v \in \mathcal{R}^{n,m}$. Hence, the first-order connectionist assumption yields

$$\boldsymbol{X}_v = \sigma\left(\sum_{k=1}^{o} \boldsymbol{A}_v(k) \cdot q_k^{-1}\boldsymbol{X}_v + \boldsymbol{B}_v \cdot \boldsymbol{U}_v\right). \tag{2}$$

Like for list processing (recurrent nets processing sequences) the output can be computed by means of $\boldsymbol{Y}_v = \sigma(\boldsymbol{C} \cdot \boldsymbol{X}_v)$. In practice, especially when dealing with graphs having high outdegree, given any node v, the state associated with its corresponding k-th child $q_k^{-1}\boldsymbol{X}_v$ can be conveniently compressed before feeding the neuron outputs. Hence, the state updating equation (2) becomes

$$\boldsymbol{X}_v = \sigma\left(\sum_{k=1}^{o} \boldsymbol{A}_k \cdot q_k^{-1}\sigma\left(\boldsymbol{P}_v \cdot \boldsymbol{X}_v\right) + \boldsymbol{B}_v \cdot \boldsymbol{U}_v\right). \tag{3}$$

where $\boldsymbol{P}_v \in \mathcal{R}^{n_r,n}$ reduces the dimension n of the pointer information from n to $n_r < n$.

Like for multilayer perceptrons operating on static data types, the learning process is carried out by optimizing the attached error function. Note that the optimization takes place by considering all the encoding neural networks by weight sharing, that is the matrices attached to the mmodes are the same for all the nodes of the graphs. The gradient computation is carried out by Backpropagation. Because of the graphical structure inherited by the encoding networks the algorithm for the gradient computation is referred to as *Backpropagation through structure*.

5 Conclusions

The extension of recurrent neural networks to the domain of directed ordered graphs reviewed in this paper is motivated by the impressive number of different application domains in which structured data seems to be more appropriate. A number of problems are still open. For instance, the elegant extension of Back-propagation through time to the case of DOAG's does not apply to more general types of graphs and the weight sharing mechanism might not be appropriate in most interesting real-world problems in which huge graphs are involved. In principle, in that case one can use the approach sketched in Figure 3, where different modules are in charge for processing different nodes.

Finally, most of the studies reviewed in paper deals with supervised learning, even though many interesting real-world problems (see e.g. iconic retrieval in image data bases) would require strong unsupervised learning capabilities of graphic objects.

References

[1] J.A. Fodor and Z.W. Pylyshyn. Connectionism and cognitive architecture: A critical analysis. *Connections and Symbols*, pages 3–72, 1989. A Cognition Special Issue.

[2] P. Frasconi, M. Gori, and A. Sperduti. A general framework for adaptive processing of data structures. *IEEE Transactions on Neural Networks*, 9(5):768–786, September 1998.

[3] C.L. Giles, G.M. Kuhn, and R.J. Williams. Dynamic recurrent neural networks: Theory and applications. *IEEE Transactions on Neural Networks*, 5(2), 1994. Special Issue.

[4] M. Gori, M. Mozer, A.C. Tsoi, and R.L. Watrous. Special issue on recurrent networks for sequence processing. *Neurocomputing*, 15(3&4), June 1997.

[5] G.E. Hinton. Special issue on connectionist symbol processing (editorial). *Artificial Intelligence*, 46(1/2), November 1990.

[6] J. B. Pollack. Recursive distributed representations. *Artificial Intelligence*, 46(1-2):77–106, 1990.

[7] A. Sperduti. Labelling recursive autoassociative memory. 6(4):429–459, 1994.