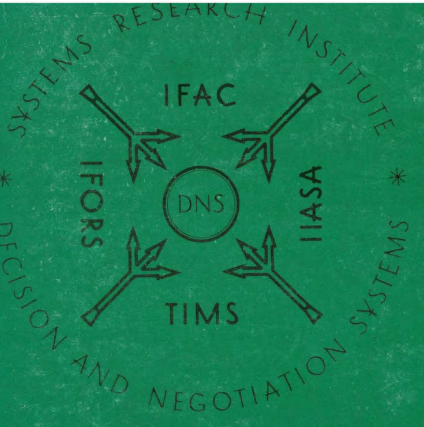

IFAC/IFORS/IIASA/TIMS

The International Federation of Automatic Control
The International Federation of Operational Research Societies
The International Institute for Applied Systems Analysis
The Institute of Management Sciences

SUPPORT SYSTEMS FOR DECISION AND NEGOTIATION PROCESSES

Preprints of the IFAC/IFORS/IIASA/TIMS Workshop
Warsaw, Poland
June 24-26, 1992

Editors:

Roman Kulikowski
Zbigniew Nahorski
Jan W. Owsiński
Andrzej Straszak

Systems Research Institute
Polish Academy of Sciences
Warsaw, Poland

VOLUME 2:

Names of first authors: **L-Z**

SYSTEMS RESEARCH INSTITUTE, POLISH ACADEMY OF SCIENCES

SUPPORT SYSTEMS FOR DECISION AND NEGOTIATION PROCESSES
Preprints, IFAC/IFORS/IIASA/TIMS Workshop, June 24-26, 1992, Warsaw, Poland

CONNECTIONIST SYSTEMS FOR THE DEVELOPMENT OF AUTONOMOUS AGENTS

Edward Szczerbicki
The Technical University of Gdansk, 80-952 Gdansk
Poland

Abstract: To support the development of an information flow between the elements of a given manufacturing agent (a group of people, machines, robots, guided vehicles) an approach is suggested that combines knowledge expressed by traditional production rules with machine learning technique based on the training of a neural network. A three layer neural configuration is used for illustrative purposes. It is demonstrated that the use of neural networks shows promise as a tool for solving management and manufacturing related problems.

Keywords: neural networks, autonomous systems, learning, information structure, manufacturing agents

1. Introduction

A three stage approach to the development of integrated multiagent manufacturing systems involves functional decomposition, agent representation, and information flow integration (Szczerbicki 1992a). Functional decompositions are modelled by AND/OR trees. Logical trees allow the analyst to generate alternative solutions in the process of formulating specifications for a given manufacturing process. Manufacturing agents represent specified functions and their development is based on the theory of autonomous systems. In Szczerbicki (1992a, b) production rules guiding the search in the functional space and development of information structures for agents are presented. Production rules are helpful in the development of traditional rule-based expert systems. In this paper an approach is discussed that allows one to develop a connectionist expert system, i.e., neural network-based expert system (Yoon et al. 1990, Gallant 1988).

For the development of information flow in autonomous systems, the knowledge domain includes many abstract concepts (Szczerbicki 1991a). The use of the back propagation algorithm to develop a knowledge base in the area of information flow illustrates the ease and appropriateness of this method for dealing with implicit knowledge and also provides a model for extension into other expert domains.

Connectionist-learning (neural network-based) algorithms have been successfully applied to many low-level cognitive tasks such as speech recognition, signal processing, character processing, character recognition, and motor control (Hecht-Neilsen 1988, Gorman and Sejnowski 1988, Rumelhart and McClelland 1986, Vemuri 1988, Maren et al. 1990).

However, the capabilities of these algorithms have rarely been explored in working with high-level cognitive processes such as information processing or manufacturing, in which traditional expert systems techniques are dominant, though the techniques have serious drawbacks in knowledge acquisition.

One of the obvious differences between a traditional expert system and a neuron-based (connectionist) expert system is a belief in how to develop intelligent systems. The traditional (rule-based) approach uses a domain expert to identify the explicit heuristics used to solve problems, whereas a neural networks approach assumes the problem-solving steps are to be derived without direct attention to how a human actually performs the task. Traditional expert systems try to figure out how the human mind is working, whereas connectionist systems mimic the most primitive mechanisms of the brain and allow the external input and output to designate the proper internal functioning.

1.1. Neurons and neural networks

A neuron has a number of branched dendrites and an axon, which are used to receive and pass information to other neurons (Knapp and Wang 1992). The neurons are connected with synapses to form a basic biocomputational network. The number of connections is so large that it provides the network with sophisticated capabilities such as logical derivation, objective perception in natural scenes, and so on.

Neural networks are loosely modelled after human networks of neurons in the brain and nervous system. They are a class of connectionist computing systems (Thornton 1991). Neural networks have long been studied in the hope of finding solutions for problems with unknown or complex internal relationships. Some examples of such studies that deal with problems in the area of manufacturing are included in Spelt et al. (1991), Burke and Rangwala (1991), Malave and Ramachandran (1991).

Adaptation, or the ability to learn, is the most important property of neural networks. A neural network can be trained to map a set of input patterns onto a corresponding set of output patterns simply by means of exposure to examples of the mapping. This training is performed by gradually adapting the internal weights of the network, so as to reduce differences between the actual network outputs (for a given set of inputs) and the desired network outputs. Neural networks which learn mappings between sets of patterns are called mapping neural networks (Chryssolouris 1990). A key property of mapping networks is their ability to produce reasonable output vectors for input patterns outside of the set of training examples (Nielsen 1987, Rumelhart et al. 1986).

2. Neural Network for Information Structure Development

The proposed development procedure includes three steps. The first step is described in other works by the author (see the reference section) and it involves the model based generation of knowledge for autonomous systems. This knowledge is expressed in the form of IF ... AND ... THEN production rules and is based on simulation of the model presented in Szczerbicki (1990, 1991b). Seventeen of such rules are included in Szczerbicki (1992a) for rule based representation of autonomous manufacturing agents. An *autonomous manufacturing agent* is a group of people, machines, robots, and/or guided vehicles that decides about its own informational requirements (autonomy) and that represents manufacturing functions (Szczerbicki 1992b).

In the second step, the rules are used to train neural network.

The third step involves the use of the trained neural network in the situations that are not covered by the rules used in the second step.

Problem-solving tasks, such as information structure development for a manufacturing agent, may be considered pattern classification tasks. The system analyst learns mappings between input patterns, consisting of characteristics of agent's external and internal environment, and output patterns, consisting of information structures to apply to these characteristics. Thus, neural networks (neural-based expert systems) offer a promising solution for automating the learning process of the analyst.

2.1. Mapping formulation, training, and use of the network

A systems analyst, while developing an information structure for a manufacturing agent, transforms certain characteristics of an agent into recommendations concerning the flow of information (Szczerbicki 1990, 1991b). These characteristics represent the input for the system and they include 5 parameters: correlation in the external environment (R), dynamics (T), interaction in the internal environment (I), delay (D), and type of the process describing the external environment (W). Output consists of two recommendations: (i) observation (or sensoring) should be present, and (ii) exchange of information should be present (the importance of the above parameters and the role of observation and exchange of information in the development of an information structure are discussed in details in Szczerbicki (1990, 1991a)). An input portion together with an output portion of the data represents a training pair. Training pairs were generated through the simulation of the formal model of manufacturing agents.

The training pairs were used to train a 5-10-2 neural network (Figure 1). The number of input nodes (5) was chosen to match the number of relevant characteristics of an agent, and

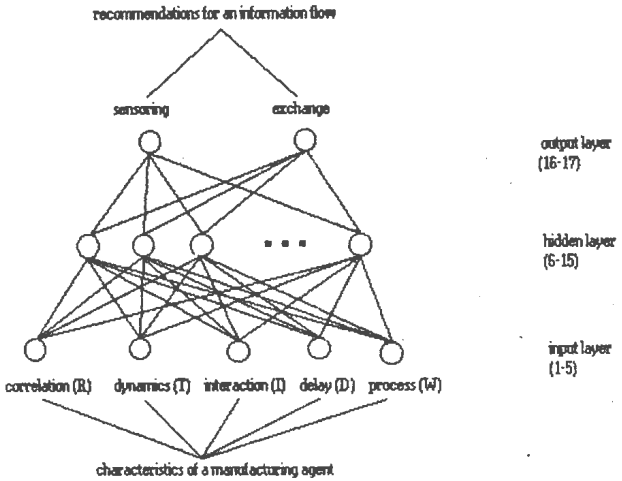

Figure 1. A knowledge-based 5-10-2 neural network

the number of output nodes (2) was chosen to match the number of information flow recommendations. The number of hidden layer nodes is not constrained in a definite way. If it is too small, the back propagation algorithm will not converge upon a set of network weights and thresholds. If it is too large it will take unnecessarily long time to converge. Since the domain production rules were known in advance, it was possible to determine an approximate lower bound on the number of hidden units needed using the guidelines set forth by Mirchandani and Cao (1989). The target values for each output node were normalized in such a way that the maximum target for each node received a value of 0.75 and the minimum target for each node received a value of 0.25. This was done to bring the target values within the output range of the sigmoid output function. The training values for each input node were identically normalized. Prior to training, the network weights were initialized to values from the interval [-1,1] and thresholds to values from the interval [-0.25,0.25]. The learning rate and momentum term of 0.9 were used in the network. These values were chosen on the basis of suggestions in the neural network literature (Rumelhart and McClelland 1986, Vemuri 1988). The network was trained with a training tolerance of 5%. Ten training pairs were used that were developed according to production rules presented in Szczerbicki (1992a, 1991b, 1990). The network was considered trained if, for all training pairs and output nodes, |(desired output - actual output)/(desired output)| < tolerance.

The back propagation (called also error back propagation) procedure (Nielsen 1987) was used for training purposes. The output error was determined by performing the forward computations in the network and comparing the results with the desired output. The sigmoid output function was assumed. The computed output error was propagated back through the network, and the weights associated with the links were changed in order to reduce the local error fraction. The momentum term was included to decrease the tendency for oscillation during the training process.

After training, additional characteristics of an agent were generated for use of the network. Testing the network's trained performance was done by the comparison of the network's recommendations concerning the flow of information with the rules that served as the basis for generating the additional characteristics. Five sets of characteristics were submitted to the network. In response, the network suggested five information flow recommendations (information structures). As an example, Table 1 presents three sets of characteristics submitted and the obtained recommendations after the trained network has been used. For the first input set (No. 1 in Table 1) the network recommends decentralized information structure. For the second, full information structure is recommended. In the third case, the network recommends routine actions without observation and exchange of information. In each case the recommendations agreed with the production rules from which the agent's characteristics were delivered.

3. Conclusion

Preliminary results of a procedure of the development of information structure for autonomous manufacturing agents employing neural network in conjunction with traditional production rules have been presented. Although the training sets were limited by the number of production rules generated so far, the procedure itself has been successfully applied and demonstrated. The approach presented shows the potential for use in real-world problems that are not intuitively straightforward. The neural network approach uses a single methodology for generating useful inferences, rather than using explicit generalization rules. Because the network only generates inferences as needed for a problem, there is no need to generate and store all possible inferences ahead of time. Further research is needed to provide a basis for the selection of a particular neural network for use in the procedure. Also, the effectiveness of the procedure for large problems, in which many information structure parameters have to be determined, remains to be investigated.

The usefulness of any traditional expert system depends on the completeness of the expert knowledge it contains. Knowledge acquisition has been identified as a major bottle-neck to

Table 1. The use of the trained network

Input characteristics			Output recommendations	
No	Value	Description	Observation (sensoring)	Exchange
1	R=0.95	strong relationship between variables describing external environment	Yes	No
	T=0	external environment is static		
	I=0.01	there is no interaction in internal environment		
	D=0	information is not delayed		
	W=0	process is independent		
2	R=0.2	weak relationship between variables describing external environment	Yes	Yes
	T=0	external environment is static		
	I=0.90	there is interaction in internal environment		
	D=0	information is not delayed		
	W=0	process is independent		
3	R=0	there is no relationship between variables describing external environment	No	No
	T=1	external environment is dynamic		
	I=1	there is interaction in internal environment		
	D=1	information is delayed		
	W=0	process is independent		

the implementation of expert system technology in many areas of engineering (Hayes-Roth et al. 1983, Siddall 1990, Mirzai 1990). Another limitation is that traditional expert system are unable to handle situations even slightly different from known prototype conditions. On the other hand, however, one can see a good opportunity of blending of traditional expert systems with neural networks. Arguments that either expert systems or neural networks should replace one another should be ignored. The potential is great for both fields, which can be pursued simultaneously to achieve the goal of intelligent behaviour.

4. References

Albus, J.S. (1991) Outline for a theory of intelligence. *IEEE Transactions on Systems, Man, and Cybernetics* 21 , no. 3, 473-509.

Burke, L.I., and Rangwala, S. (1991) Tool condition monitoring in metal cutting: a neural network approach. *Journal of Intelligent Manufacturing* 2, no. 5, 269-280.

Chryssolouris, G., Lee, M., Pierce, J., and Domroese, M. (1990) Use of neural networks for the design of manufacturing systems. *Manufacturing Review* 3, no.3, 187-194.

Gallant, S.I. (1988) Connectionist expert system. *Communications of the ACM* 31, no.2, 152-169.

Gorman, R.P., and Sejnowski, T.J. (1988) Analysis of hidden units in a layered network trained to classify sonar targets. *Journal of Neural Networks* 1, no.1, 75-89.

Hayes-Roth, F., Waterman, D., and Lenat, D.B. (1983) *Building expert systems* Addison-Wesley, Reading.

Hecht-Neilsen, R. (1988) Application of counter-propagation network. *Journal of Neural Networks* 1, no.2, 131-139.

Knapp, G.M., Wang, H-S.B. (1992) Neural networks in acquisition of manufacturing knowledge. In *Intelligent Design and Manufacturing* Kusiak, A. (ed.) Wiley, New York. 723-744.

Malave, C.O., and Ramachandran, S. (1991) Neural network-based design of cellular manufacturing systems. *Journal of Intelligent Manufacturing* 2 , no.5, 305-314.

Maren, J.A., Harston, C.T., and Pap, R.M. (1990) *Handbook of neural computing applications* Academic Press, New York.

Mirchandani, G., and Cao, W. (1989) On hidden nodes for neural nets. *IEEE Transactions*

on Circuits and Systems, 36, no. 5, 661-664.

Mirzai, A.R. (1990) *Artificial intelligence. Concepts and applications in engineering.* Chapman and Hall Computing, London.

Nielsen, H. (1987) Counterpropagation networks. *Proceeding of IEEE First International Conference on Neural Networks*,19-22.

Rumelhart, D.E., Hinton, G.E. and Williams, R.J. (1986) Learning internal representation by error propagation. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition.* Rumelhart, D.E., and McClelland, J.L. (ed) MIT Press, Cambridge MA, 318-362.

Rumelhart, D.E. and McClelland, J.L. (1986) *Parallel distributed processing: exploration in the microstructure of cognition* MIT Press, Cambridge MA.

Siddall, J.N., (1990) *Expert systems for engineers* Marcel Dekker, New York.

Spelt, P.F., Knee, H.E., and Glover, C.V. (1991) Hybrid artificial intelligence architecture for diagnosis and decision-making in manufacturing. *Journal of Intelligent Manufacturing*.2 , no. 5, 261-268.

Szczerbicki, E. (1990) Autonomous groups functioning - the role of correlation and interaction. *International Journal of Systems Science*, 21, no. 10, 2037-2047.

Szczerbicki, E. (1991a) Structuring an information flow for autonomous systems. *International Journal of Systems Science*, 22, no.12, 2599-2609.

Szczerbicki, E. (1991b) Information flow evaluation in autonomous groups functioning. *IEEE Transactions on Systems, Man, and Cybernetics*, 21, no. 2, 402-408.

Szczerbicki, E. (1992a) Rule-based functional decomposition and representation of manufacturing agents. *Systems Analysis, Modelling, Simulation*, 9, (in press).

Szczerbicki, E. (1992b) Rule-based integration of autonomous multiagent systems. *International Journal of Systems Science* 23, (in press).

Thornton, C.J. (1991) Introduction to connectionist computing. *CAD* 23, no. 8, 530-538.

Vemuri, V. (1988) *Artificial neural networks. Theoretical concepts.* MIT Press, Cambridge.

Yoon, Y.O., Brobst, R.W., Bergstreser, P.R., and Peterson, L.L. (1990) A connectionist expert system for dermatology diagnosis. *Expert Systems*,1, no.4, 22-31.