REPORT DOCUMEN

AD-A257 215

orm Approved
OMB No. 0704-0188

D

1. AGENCY USE ONLY (Leave blank)	2. REPORT 1991	OVERED Final Technical Report

4. TITLE AND SUBTITLE Integrating Neural Network and Expert Reasoning: An Example	5. FUNDING NUMBERS N0014-88-K-0560
6. AUTHOR(S) James Hendler & Leonard Dickens	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) University of Maryland Computer Science Department College Park, MD 20742	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING MONITORING AGENCY NAME(S) AND ADDRESS(ES) ONR Defense Technical Information Center Defense Logistics Agency Cameron Station, Alexandria, VA 22304-6145	10. SPONSORING MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

DTIC ELECTE NOV 04 1992 S D A

12a. DISTRIBUTION / AVAILABILITY STATEMENT This document has been approved for public release and sale; its distribution is unlimited.	12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

In this paper we discribe a shell which has been developed to allow an integration of neural network and expert systems technology. The system, SCRuFFy, is based on an analysis of the different abilities and time courses of NN and AI systems. Critical to the processing of this system is a temporal pattern matcher which is used to mediate between the two subsystems, providing a "signal to symbol" mapping. This mapping allows the expert system to reason abou the time course of signals which are classified by a connectionist network which is trained via classical back-propagation of error during a separate training phase. An example of the simulated control of the temperature of an underwater welding robot is presented to demonstrate these capabilities.

92 11 072

409022

92-28737

r 36

14. SUBJECT TERMS Neural nets Expert Systems Artificial Intelligence	15. NUMBER OF PAGES
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT

To appear: AISB Conference, April 1991.

Integrating Neural Network and Expert Reasoning: An Example

James Hendler[1]
Leonard Dickens

Department of Computer Science
University of Maryland
College Park, MD 20742

ABSTRACT

In this paper we describe a shell which has been developed to allow an integration of neural network and expert systems technology. The system, SCRuFFy, is based on an analysis of the different abilities and time courses of NN and AI systems. Critical to the processing of this system is a temporal pattern matcher which is used to mediate between the two subsystems, providing a "signal to symbol" mapping. This mapping allows the expert system to reason about the time course of signals which are classified by a connectionist netwo:! which is trained via classical back-propagation of error during a separate training phase. An example of the simulated control of the temperature of an underwater welding robot is presented to demonstrate these capabilities.

INTRODUCTION

Steels (1989) has pointed out that connectionist networks can perform heuristic classification tasks in a matter different than, but comparable to, the heuristic classification used in many expert systems. He argues that the distinction of which to use should be based on an analysis of features of the domain, rather than based on pre-theoretical biases. In addition he outlines a set of criteria to determine those situations in which a statistically-based classification method (such as the connectionist back-propagation algorithm) can be used: the cost of observation should be low, the features of the class may or may not all be present, the set of classes is known prior to classification time, the categorizations are based on statistical criteria, and hierarchical structure of the classes is either not necessary or used in only a limited manner.

One type of operation fitting these criteria is the categorization of sensor signals into classes. Typically, sensor data is obtained cheaply at a high rate of speed, features of the class may be underdetermined (necessitating a learning algorithm), classification of the signals is desired to match known categories, classification is based on high-order statistical correlations between features in the signal, and no hierarchical categorization of the signals is typically performed. Thus, it is no surprise that one of the greatest successes of connectionist models is in the classification of sensor signals.

Making intelligent decisions based on signals received from sensors,however,

[1]e-mail: hendler@cs.umd.edu. Dr. Hendler is supported in part by ONR grant N00014-88-K-0560 and NSF grant IRI-8907890 and by the UM Systems Research Center.

does not fit these criteria at all. Such decisions may require merging a long time history of sensor results, all features of a situation may need to be present before expensive corrective operations would be authorized, the set of all possible problems may not be known, the recognition of a problem is often context dependent, and hierarchical structure is often used in the decision making processes. Thus, the connectionist approach seems less suited to such tasks than to the classification tasks themselves, and such decision making remains the domain of traditional AI systems.

Unfortunately, this dichotomy of utility leads to a major problem for those who wish to work in domains such as those for the control of mechanical systems in which sensor feedback is needed to monitor the status of ongoing processes. For example, given an automated robot working in a hostile domain (for example an underwater welder or space based system), the controller needs to interpret sensor signals to determine whether normal operating conditions are being observed, or whether some error condition is occuring. If the latter, corrective action needs to be taken – perhaps adjusting the system or even shutting it down to prevent serious damage. As argued above, this sort of application requires the classification of sensor signals, deliverable using the connectionist framework, and symbolic decision making, not well suited to that approach.

The differences in the computing metaphors underlying symbolic AI and connectionist systems have led researchers to conclude that these approaches are either separate paradigms (Smolenksy, 1990) or fill different computational niches (Dyer, 1988). Faced with developing systems needing multiple capabilities, research has generally focused on drastically extending the capabilities of one sort of reasoning or the other. This includes efforts to produce symbolic reasoning using connectionist systems (cf. Shastri, 1985) or to use traditional AI techniques to handle the sorts of perceptual tasks currently solved most successfully by connectionist networks[2].

In this paper we describe an alternative: a system which integrates a connectionist classifier and a symbolic reasoner. Based on an analysis of the differing criteria for heuristic classifiers and complex decision makers, we have targeted a hybrid system as the best way of developing applications such as the sensor-based control systems described above. To facilitate the near-term development of such systems we have been working on the development of an applications-oriented shell providing an integration of a connectionist back propagation learning network with an OPS5-based expert system development language. In this paper we describe the architecture of our system, give a simple example of its use, and describe extensions to the prototype shell which are currently uderway.

THE *SCRuFFy* HYBRID SYSTEM SHELL

We have developed the SCRuFFy[3] system as a prototype shell for the development of applications which require the merging of connectionist and symbolic techniques, such as the control systems described in the introduction. A diagram

[2]As an example of this, consider the work in AI models of machine vision, which often involve pushing symbolic reasoning to the pixel level (cf. Canning et.al., 1987).

[3]SCRuFFy: *S*ignals *C*onnections and *Ru*les - *F*un *F*or ever*y*one.

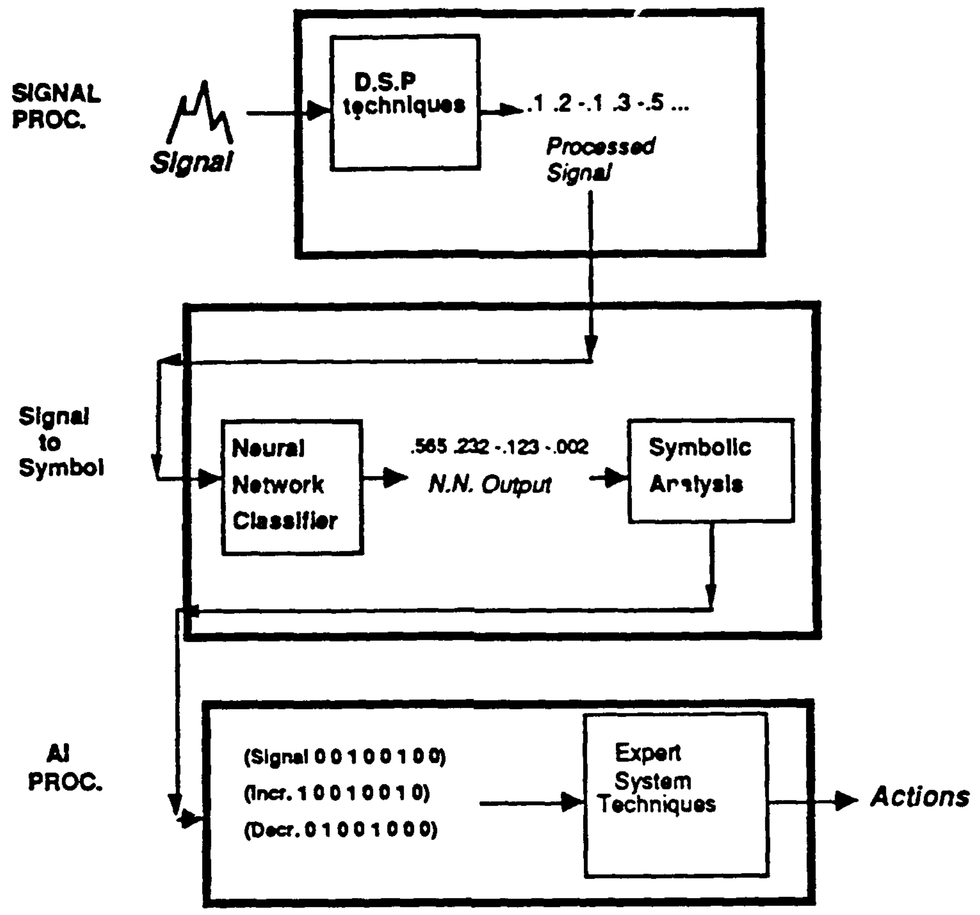

Figure 1: Flow of Control in SCRuFFy

of the flow of control in the system is shown in Figure 1. A sensor signal feeds through a digital signal processing program which converts it into a vector of discrete values. A set of known signals is gathered, and these are used to train a back propagation network to perform the signal classification task. Once the network is trained, the flow of control is for a new signal to be received, to be sampled, and to be classified using the connectionist network. The output of this process is then fed into a symbolic analysis program - a program which tracks the changes in signal classification over time and produces symbolic information describing the time course of the signal. This symbolic information is put onto a blackboard (or working memory) and thus a traditional blackboard-based expert reasoning system can make control decisions based on this information.

The heart of the SCRuFFy system is a "temporal pattern matcher", which provides an interface between the continuous signal sampling and classification, provided by the analog connectionist system, and the expert system used in making the control decisions. The pattern matcher is based on the observation that the particular activation patterns from the connectionist component for a given signal will usually be less interesting to the expert system-based reasoning than will the pattern of a set of signals over time. For example, in a simple application, we might have 10 outputs coming from the connectionist system. Two of these might be indicators of normal operation, while the 8 others represent different kinds of errors. If a controller takes no action until one of the error signals is the highest output, then this may result in drastic corrective actions needing to be taken. However, a recognition that one of the errors is increasing over time, even though it is not currently the highest value, could allow a less drastic response.

We have based our patterns on the temporal relations discussed by Allen and Koomen (1983). These include relations such as "before, after, during, etc." which are specified for temporal intervals. These patterns are used to reason about the time course of a set of signal samples, rather than a single input. The heart of the matcher is an array, which keeps track of sample values for each of the outputs produced by the connectionist classifier. (The number of past outputs stored is a user specifiable parameter.) The matcher then examines its rules to see if any of them have conditions which are satisfied[4].

Rules for the matcher are specified over intervals using predicates. For example, a rule can check for the predicate "increasing" over an interval of time. When the set of predicates are all matched, a set of associated actions can be taken. Variables can be bound, thus providing a standard production-rule-like method for associating the patterns and actions. In addition, the actions taken can include posting items on the blackboard used by the expert system. For example, the rule:

```
(R carbon-leakage
   (highest SIGNAL-B)
   (decreasing (recent ?Sigtype))
   (not (eq ?Sigtype SIGNAL-B))
 -->
   (format t "~%Carbon leakage noted!")
   (post-to-blackboard NOTED CARBON-LEAKAGE <current-time>))
```

could be used to recognize a situation where the connectionist network produces a set of outputs where signal type "B" is the highest, but a "non-B" signal is decreasing over a small set of samples[5]. When this is recognized two actions are taken: a message is printed out, and a working memory element indicating carbon leakage is posted for the expert system.

EXAMPLE

As an example of the use of SCRuFFy, we have developed a very simple expert system for the control of an underwater welding robot using information from an acoustic sensor. The system is fed an acoustic signal (which is randomly supplied via a simulator which uses actual signals sampled from a 1MHz sensor). This acoustic signal is analyzed by the back propagation network to produce a four element vector, representing the relative classification strengths of normal welding and three known error conditions. When the system notices an increasing strength in one of the error signals, even if it is not the highest component of the classification vector, it can suggest repair actions to be taken, such as reducing the temperature or shutting down the welder. Actual control of a complex system has

[4]A Rete-like algorithm can be used for efficiency, although our current implementation uses a simpler matching procedure.

[5]The interval "recent" is used to indicate a number of samples set by the user as a parameter.

not yet been attempted, instead, diagnostic messages describing what is occuring and what actions are suggested are the outputs of the system.

The following is an example of SCRuFFy running on acoustic data. Comments are indicated in *italics*, output from the expert system component (i.e. system control decisions) are indicated by SMALL UPPERCASE text. The example uses signals from a simulator designed to show particular behaviors. In this case, we see SCRuFFy taking actions to control "streaming," a particular error condition arising in the welding.

```
; The expert system is initialized

 ********** INITIALIZING EXPERT SYSTEM
 ********** INPUT WATER TEMPERATURE: 021
 ********** INPUT WATER DEPTH: 030
; the context information provided by water depth and temperature
; is used by the SCRuFFy's expert system component.
=============================================
; the first signal is sampled
Signal (after DSP) is:

 0.727 0.672 0.508 0.738 0.56 0.633 0.144 0.429 0.196 0.554 0.658 0.483
 0.602 0.41 0.371 0.955 0.225 0.796 0.021 0.922 0.447 0.255 0.549 0.855
 0.97 0.807 0.77 6 0.641 0.834 0.674 0.581 0.726 0.005 0.716 0.073
 0.664 0.69 0.191 0.905 0.367 0.427 0.844 0.783 0.078 0.403 0.453 0.775
 0.989 0.292 0.726 0.355 0.335 0.61 0.12 0.723 0.629 0.857 0.889 0.864
 0.63 0.369 0.406 0.246 0.617 0.552 0.014 0.09 3 0.633 0.704 0.682
 0.592 0.615 0.769 0.945 0.893 0.512 0.68 0.587 0.473 0.681 0.732 0.475
 0.956 0.142 0.986 0.947 0.827 0.82 0.602 0.764 0.209 0.884 0.788 0.208
 0.301 0.036 0.304 0.351 0.347 0.249 0.124 0.352 0.922 0.272 0.618
 0.768 0.92 7 0.554 0.925 0.057 0.321 0.09 0.171 0.527 0.86 0.231
 0.768 0.111 0.798 0.78 0.891 0.786 0.905 0.678 0.751 0.981 0.957 0.732
; This signal is input to the pre-trained connectionist network
Output from Connectionist network is:
 .9296 .0145 .0759 .0465
; And the output is recorded for use by the temporal pattern matcher

 The temporal pattern matcher finds any patterns that match
Output from temporal pattern matcher:
  Normal operation.
  A is highest, and no increasing minors
; Control is then transferred to the expert systems
 Transferring control to Expert System:
 __________
 ********** NORMAL OPERATION OF WELDER
; and diagnostic or control messages are produced
=============================================
 ... ; Several normal signals are omitted
=============================================
; another signal is sampled
Signal (after DSP) is:
   ...  ; signals are deleted to save space
Output from Connectionist network is:
  .9405 .0323 0.005 .2729
Output from temporal pattern matcher:
  Normal operation.
 A is highest, but increasing error signal B
; this time, a temporal pattern fires representing an increase in
; an error signal
 Transferring control to Expert System:
 ----------
 ********** EVIDENCE OF STREAMING INCREASING
 ********** CONTROL: NO ACTION TO BE TAKEN AT THIS TIME
; Given the error rates of neural network processing, it is
; possible that this is network error, so we wait for another
; signal to confirm the error.
==========================================
; The next signal
Signal (after DSP) is:
 ...
Output from Connectionist network is:
.9227 .2279 .0311 .0148
Output from temporal pattern matcher:
 Normal operation.
 A is highest, but increasing error signal B
 Transferring control to Expert System:
 ----------
 ********** EVIDENCE OF STREAMING INCREASING PERSISTENTLY
 ********** CONTROL: DECREASE TEMPERATURE (MINOR = .2)
 ********** CHANGING WELDING TEMPERATURE: WAS 40 - NOW 39.8
;this time, an action is taken, since the error is repeated.
==========================================
Signal (after DSP) is:
 ...
Output from Connectionist network is:
.4158 .5424 .2176 .0037
Output from temporal pattern matcher:
 Time 1010:
 Channel B is highest.
 Transferring control to Expert System:
 ----------
 ********** NON-OPERATING SIGNAL TYPE STREAMING ENCOUNTERED
 ********** ERROR OCCURS AFTER INCREASE
  ; it is recognized that this is not a transient - since previous
  ; action was taken for this type of error.
 ********** CONTROL: DECREASE TEMPERATURE (MAJOR = 2.5)
 ********** CHANGING WELDING TEMPERATURE: WAS 39.8 - NOW 37.3
  ; so a more drastic control measure is suggested by the
  ; expert reasoner.
=========================================
```

FUTURE WORK

The hybrid system described in this paper is based primarily on an analysis of the differing capabilities of the connectionist and symbolic approaches, and on the different temporal properties of continuous and discrete systems. The temporal pattern matcher provides an interface between these components, allowing a smooth integration. Such components are clearly necessary to any systems integration of these technologies. In the longer term, however, the design of our current system, providing only a "feed-forward" model of computation (no feedback loops) is considered to be quite a limitation. The overall system is a "loosely-coupled" hybrid model[6] and cannot take self-modifying actions.

The current system provides a mapping from signals through to "control" actions proposed by the expert system reasoning component. There is no reason that these control actions could not be self-modifying, that is, changing the various parameters and rules used by the system itself. Thus, in addition to control of an external system, the current model can be extended for self control. The major forms of self-modification we are currently examining include:

- Changing the temporal patterns: Upon the receipt of external information (or via internal decisions), the system can modify the set of temporal patterns of interest.

- Changing network parameters and/or output multipliers: Neural networks are governed by a set of parameters (learning rate, momentum, etc.). Controlling these parameters dynamically is an area just starting to be explored. As control models are developed, "intelligent" control (using the expert system) can be implemented, and experimented with, in this framework.

- Changing network configuration: Certain sorts of network models (such as Hopfield networks for minimal cost paths) have structural constraints depending on external information (path cost between nodes, etc.) When information is discovered during processing, it may reflect a change in this information, and thus necessitate dynamic reconfiguration of the network.

In addition, the current model provides for a single network reporting to a single temporal pattern-matcher. However, the blackboard architecture on which SCRuFFy is based provides a natural mechanism for the integration of multiple knowledge sources. A hierarchical model of the temporal patterns (in which patterns found for a given network can be compared with other patterns for other networks) can provide for a merging of the data from various sensors. By exploring the use of such patterns, coupled with the parameter changes described above, we believe this framework will allow for experimentation with varying models of multi-sensor systems.

CONCLUSIONS

In this paper we have described a shell which was developed to allow an integration of neural network and expert systems technology. The system, SCRuFFy, is based on an analysis of the different abilities and time courses of neural network and AI systems. Critical to the processing of this system is a temporal pattern matcher which is used to mediate between the two subsystems, providing a "signal to symbol" mapping. This mapping allows the expert system to reason about the time course of signals which are classified by a connectionist network which is trained via classical back-propagation of error during a separate training phase. An example of the simulated control of the temperature of an underwater welding robot was presented to demonstrate these capabilities.

[6]See (Hendler, 1989b) for a discussion of loosely and tightly coupled hybrid models.

As a final, more "philosophical" note, we wish to point out that the design and motivation of the SCRuFFy system derives largely from a careful analyis of the problem-solving exhibited by connectionist and symbolic systems (Steels, 1989; Hendler, 1989b). The techniques used in this analysis were not special to either approach, but rather derived from examining the information processing features of the two so-called paradigms. Such an analysis, common in the expert systems community, but often ignored by neural network developers, enables the strengths and weaknesses of the approaches to be identified, and allows the design of component based systems, such as SCRuFFy, which can exploit the appropriate technologies for the particular aspects of problems which are being solved. Thus, in short, we agree strongly with Steels in his contention that "when this analysis is made ... the differences between expert systems and connectionist problem solving are relatively marginal as far as operation is concerned." By viewing the technologies in this light, it is easier to see that they can be viewed as cooperating, rather than competing, approaches.

REFERENCES

[1] Allen, J.F. and Koomen, J.A. Planning using a temporal world model *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*, 1983.

[2] Canning, J., Kim, J., and Rosenfeld, A. Symbolic pixel labeling for curvilinear feature detection, *Proc. DARPA Image Understanding Workshop*, Morgan Kaufmann Publishers, Feb., 1987.

[3] Dyer, M. *Symbolic NeuroEngineering for Natural Language Processing: A Multilevel Research Approach* Technical Report UCLA-AI-88-14,Computer Science Dept., University of California, Los Angeles 1988.

[4] Hendler, J.Marker-passing over microfeatures: Towards a hybrid symbolic/connectionist model *Cognitive Science*, 13(1), March, 1989a.

[5] Hendler, J. Problem Solving and Reasoning: A Connectionist Perspective, in *Connectionism in Perspective*, R. Pfeifer, Z. Schreter, and L. Steels (eds.) Amsterdam: Elsevier, 1989b.

[6] Smolensky, P. Connectionist AI, Symbolic AI, and the Brain. *AI Review*, 1990.

[7] Shastri, L. *Evidential Reasoning in Semantic Networks: A formal theory and its parallel implementation* Doctoral Dissertation, Computer Science Department, University of Rochester, Sept., 1985.

[8] Steels, L. Connectionist Problem Solving – An AI Perspective, in *Connectionism in Perspective*, R. Pfeifer, Z. Schreter, and L. Steels (eds.) Amsterdam: Elsevier, 1989.