https://ntrs.nasa.gov/search.jsp?R=19910012475 2018-05-20T11:59:41+00:00Z

N91-21788

Using Fuzzy Logic to Integrate Neural Networks and Knowledge-based Systems

John Yen
Department of Computer Science
Texas A&M University
College Station, TX 77843
(409) 845-5466

April 9, 1990

Abstract

Even though the technology of neural nets has been successfully applied to image analysis, signal processing, and pattern recognition, most real world problems are too complex to be solved purely by neural networks. Two important issues regarding the application of neural networks to complex problems are (1) the integration of neural computing and symbolic reasoning, and (2) the monitoring and control of neural networks. Most hybrid models attempt to integrate neural net and symbolic processing technologies at the level of basic data representation and data manipulation mechanisms. However, intrinsic differences in the low-level data processing of the two technologies limit the effectiveness of that approach. This paper discusses the role of fuzzy logic in a hybrid architecture that combines the two technologies at a higher, *functional* level. Fuzzy inference rules are used to make plausible inference by combining symbolic information with soft data generated by neural nets. Neural networks are viewed as modules that perform flexible classification from low-level sensor data. The symbolic system provides a global shared knowledge base for communications and a set of *control tasks* for object-oriented interface between neural network modules and the symbolic system. Fuzzy action rules are used to detect situations under which certain control tasks need to be invoked for neural network modules. The hybrid architecture, which supports communication and control across multiple cooperative neural nets through the use of fuzzy rules, enables the construction of modular, flexible, and extensible intelligent systems, reduces the effort for developing and maintaining such systems, and facilitates their application to complex real world problems that need to perform low-level data classification as well as high-level problem solving in the presence of uncertainty and incomplete information.

1 Introduction

Recent development of neural network technology has demonstrated many promising applications in the areas of pattern recognition, image processing, and speech recognition. However, most real world problems are too complex to be solved purely by current neural network technologies. This paper addresses two important issues regarding building complex intelligent computer systems based on neural networks.

1. *How to integrate neural computing with symbolic reasoning?*
   A complex application usually can benefit from a synergistic integration of neural computing and symbolic reasoning. For example, in anti-submarine warfare, one might like to combine signal processing results computed in a neural net with symbolic analyses of evidence such as database information (e.g., records of confirmed vessel departures from port) and extended inference procedures (e.g., hypotheses about plausible mission plans). Many other problems, ranging from speech and vision to space applications, share this property of needing synergy between neural nets and symbolic approaches.

2. *How to monitor and control the behavior of neural networks?*
   If one wishes to construct a real world application such as anti-submarine warfare using neural networks, it is crucial to have mechanisms for interpreting and reacting to the results produced by the neural nets, so that the overall system can cope with the rapidly changing and unanticipated situations. For example, after being activated by an input pattern, a bidirectional associative memory, or BAM[11], might converge to a pattern not belonging to the set of training patterns. This misclassification phenomenon can be caused by having overly similar or numerous training patterns. In either case, the BAM needs to be modified (i.e., certain training patterns need to be removed from the training set) to improve its performance. Therefore, the system needs a *controller* that oversees the behavior of the neural networks. A general mechanism that supports the control across multiple cooperative neural nets will enable the construction of modular, flexible, and extensible neural net systems, reduce the effort for developing and maintaining such systems, and facilitate their application to complex real world problems. The need of a higher-level system for evaluating the performance of neural networks has also been suggested by other researchers [14].

This paper discusses the role of fuzzy logic in integrating neural networks and symbolic systems and in supervising the behavior of neural networks. To do this, we propose a hybrid architecture that uses fuzzy logic to combine the two technologies at a higher, *functional* level. Two types of fuzzy rules are supported by the architecture: fuzzy inference rules and fuzzy action rules. Fuzzy inference rules are used to assimilate the outputs of neural nets, which are often soft data [24], into the symbolic system. Fuzzy action rules are used to issue control tasks, which are implemented by methods in object-oriented programming, for activating, training, and modifying neural nets. Neural networks are viewed as modules that perform flexible classification. The symbolic system provides a global shared knowledge base for communications and a fuzzy rule interpreter for performing rule-based reasoning.

Most hybrid models attempt to integrate neural net and symbolic processing technologies at the level of basic data representation and data manipulation mechanisms. However, intrinsic differences in the low-level data processing of the two technologies limit the effectiveness of that approach. In contrast, our approach combines the two technologies at a higher, *functional* level. The symbolic system views neural networks as modules that (1) extend its reasoning capabilities into flexible classification and data associations, and (2) extend its learning capabilities into adaptive learning. Neural nets each view the symbolic system as providing a global shared memory for communications and a controller, built using fuzzy action rules, for activating, training, and monitoring them. Fuzzy inference rules are used to pass data between the two subsystems; and fuzzy action rules are used to pass action between the two.

The key features of the proposed architecture that will provide these desirable properties include the following:

1. Fuzzy rules can invoke neural nets for testing "soft" (fuzzy) conditions in their left-hand-sides.

2. Recognition of situations requiring actions on neural networks is accomplished via fuzzy action rules, whose actions are modified by the degree that the rules' conditions are matched.

3. Both high-level descriptions (e.g., input-output characterizations) and the behavior (e.g., performance evaluations) of neural networks will be modeled using a principled frame-based language.

4. The symbolic system will interact with neural nets through a set of generic functions called *control tasks*. Control tasks will be implemented using methods in object-oriented programming so that common methods can be shared, and specific methods can override general ones.

In the following sections, we first discuss the background of this work, then we describe the hybrid architecture with an emphasis on the features mentioned above. Finally, we summarize the benefits of our approach.

2 Background

2.1 Two Complementary Technologies: Neural Networks and Artificial Intelligence

Neural networks and symbolic reasoning are two complementary approaches for achieving the same goal: building autonomous intelligent systems. The major strengths of Neural Networks are their capabilities for performing flexible classification and adaptive learning. By automatically capturing similarities among training instances (i.e., adaptive learning), neural networks are often able to perform flexible classification. That is, when given input data which is similar, but not identical, to inputs upon which the system has been trained, the network generates output similar to the trained responses. Consequently, a trained neural network is able to classify data approximately even when that data is incomplete or noisy. Thus, while most AI systems cannot tolerate such data, neural networks promise a system whose performance *gracefully degrades* under those circumstances.

On the other hand, neural networks have several major weaknesses. They have trouble handling multiple instances of the same concept. Viewed as a pattern-matcher, they have trouble dealing with patterns containing variables. They tend to be specialized for a specific task. Solving complex tasks is likely to require cooperation between many neural networks, but managing their intercommunication is not well-understood. Control of the activation and learning behavior of these networks by higher-level modules is also not well-understood. Because their internal representation is in a form that cannot be comprehended by the user easily, it is hard to explain the rationale behind the output of neural networks. Although some of these problems have been addressed by neural network researchers (e.g., schema theory[2] addresses the first two issues), a neural net approach that addresses all these problems is yet to be developed. The goal of this research is to develop a comprehensive solution to these concerns using fuzzy logic and existing AI techniques.

Certain AI techniques suggest solutions to the problems illustrated above. Different instances of a concept are easily represented using frame-based knowledge representation systems. Variables often occur in patterns, which can be matched with data using a pattern matching facility. The notion of supporting many independent modules that communicate through a *global knowledge base* accessible to all modules is an idea central to many AI systems. For example, blackboard architectures maintain a data structure (the "blackboard") where all knowledge sources can post or retrieve information. Production system architectures also have a working memory that all productions match their conditions against and act upon. An AI system may also provide a higher-level controller, often called the meta-level architecture, that has knowledge about the lower-level system and is able to control the lower-level system in various ways. The explanation capabilities of AI systems have been enhanced by explicitly representing problem solving strategies [15].

Our integration of AI capabilities with neural nets is designed to address these issues. In Section 2.2, we explain the concerns driving the design. In Section 3, we detail our approach.

2.2 Problems with Current Hybrid Approaches

Combining neural networks and AI is certainly not a new idea, but previous efforts have not addressed the important issues raised above. A number of researchers have used neural networks to reimplement AI techniques such as production systems and semantic networks [19, 7]. Work in this area mainly demonstrates what neural networks can do, not that their implementations are better than the conventional ones. Others have applied neural networks to expert systems, natural language understanding, and other areas that have mainly utilized conventional AI techniques[9]. Work in these first two categories applies current neural net technologies, rather than addressing weaknesses of neural nets. Furthermore, it has demonstrated neural net implementations of things that AI can easily handle, rather than things that AI has great difficulties in doing (e.g., partial matching). A few researchers have introduced ideas from neural networks into conventional AI techniques or architectures. For example, Anderson's ACT^* architecture incorporates the notion of "activation values" into the memory structure and the rule base of a production system architecture [1]. Although such hybrid models do attempt to augment the weaknesses of AI, they do not attempt to address issues regarding multiple neural nets because there are no neural net modules in these connectionist models at all. Finally, some efforts have introduced ideas from AI into neural nets. Network regions, for instance, impose hierarchical structures from frame-based systems onto neural networks[6]. Although concerned with the weakness of neural nets, these efforts have not been able to overcome the two technologies' intrinsic differences in data representation and data manipulation mechanisms.

In neural networks, data are represented in a distributed fashion within dynamic networks and data manipulation involves numeric computations. In artificial intelligence, each conceptual entity is represented as a unit composed of symbols and pointers to other units, and data manipulation involves logical deduction and pattern matching. Our approach to this mismatch of representations is to integrate AI, not with these basic mechanisms of neural networks, but rather with their high-level functions: i.e., *classification* and *data association*. These refer to the capability of a neural net to take an input pattern and either classify it with respect to some set of classes, or generate an output pattern most closely associated with the input pattern. Viewed at this functional level, these capabilities are closely related to pattern matching and automated reasoning functions in symbolic systems.

Based on these observations, we will describe a novel hybrid architecture that alleviates the difficulties encountered by current hybrid models through the use of fuzzy logic in integrating the two paradigms at their functional levels. The architecture provides an extremely high degree of synergy between the approaches, along precisely the dimensions required to facilitate ease of programming and enable scaling-up to larger problems.

2.3 Fuzzy Logic and Neural Networks

Several techniques for integrating fuzzy logic and neural networks have been suggested. For instance, neural nets have been suggested for learning the membership functions of a fuzzy set [16]. The learning techniques in neural nets have been applied to learning fuzzy control rules [12]. Finally, fuzzy cognitive map suggests an approach for capturing fuzzy knowledge within the framework of associative memories [10]. Our discussion here will be focused on the roles of fuzzy logic in integrating multiple neural networks and knowledge-based systems and in monitoring the performance of neural networks.

3 A Hybrid Architecture

A high-level block diagram of the proposed hybrid architecture is shown in Figure 1. The architecture has four major components: (1) a set of neural net modules, (2) a symbolic system consisting of a global knowledge base, (3) a fuzzy rule system that supports fuzzy inference rules and fuzzy action rules, (4) and an object-oriented interface between the symbolic system and the neural nets. The neural nets process data obtained either from external sensor devices or from the knowledge base of the symbolic system. The global knowledge base consists of a fuzzy database and a *neural-network taxonomy* that describes meta-level knowledge about the neural nets themselves. The fuzzy database stores data and hypotheses that can be uncertain, imprecise, or vague. The neural-net taxonomy consists of *neural-net classes*, (shown as circles in Figure 1) and individual *neural-net objects* that form the leaves of the taxonomy (shown as rectangles). For instance, the neural-net object BAM_1 belongs to the neural net class **BAM** (Bidirectional Associative Memory), and inherits all the general properties (e.g., its training procedure and its activation process) of the **BAM** class. There is one neural-net object for each neural net module. The fuzzy rule base consists of two types of rules: *fuzzy inference rules* and *fuzzy action rules*. Fuzzy inference rules make plausible inferences by combining symbolic

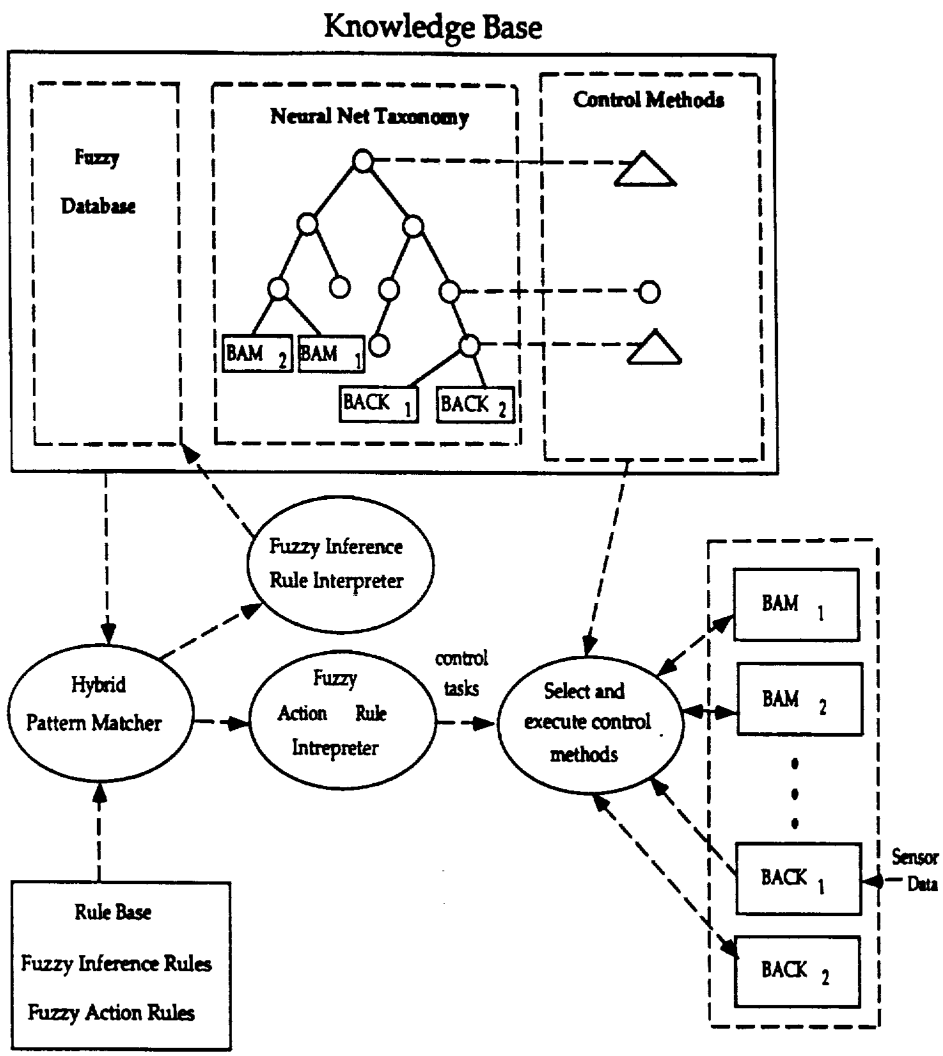

Figure 1: The Hybrid Architecture

information with the outputs of neural networks. *Control tasks* can be invoked either by procedure calls or by fuzzy action rules to effect activation, learning, and modification of neural networks. These control tasks are performed by selecting and executing methods that are inherited through the neural network taxonomy.

The hybrid architecture is an extension of CLASP [23], an advanced AI programming environment that fuses the best aspects of frames, rules, and object-oriented programming. In the following sections, we discuss four major technical issues of the proposed hybrid architecture:

1. Using fuzzy inference rules to combine the output of multiple neural networks with symbolic information;
2. Modeling meta-level knowledge about neural networks in a symbolic knowledge base;
3. Using a set of *control tasks*, which are implemented by methods in object-oriented programming, to define the interface between symbolic systems and neural nets;
4. Using fuzzy action rules to recognize situations necessitating actions upon neural networks.

Throughout the following discussion, we will use a sensor fusion system for anti-submarine warfare as an example to illustrate our approach. This hypothetical system consists of multiple neural nets for classifying various kinds of sensor input and for integrating various information about submarines, along with a symbolic expert system for analyzing the findings and planning anti-submarine strategies.

3.1 Fuzzy Inference Rules

We use fuzzy inference rules to assimilate the outputs of neural networks into the symbolic system, because neural networks often generate classification results that are imprecise in nature. For instance, a neural network that determines the hostility classification of a submarine could generate a qualitative measure of hostility (e.g., hostility degree is 0.7), or a membership values of several fuzzy sets (e.g., membership value of very-hostile is 0.6, membership value of hostile is 0.8, ...).

A fuzzy inference rule checks certain soft conditions, than make a plausible conclusion based on the degree those conditions are satisfied. The condition side of a fuzzy rule consists of fuzzy conditions as well as non-fuzzy condition. A fuzzy condition can be checked by invoking a neural net module in a data-driven fashion (i.e., the neural net

```
If Source was lost due to fade-out in the NEAR-PAST, and
   Similar source started up in an another frequency, and
   Locations of the sources are relatively CLOSE
Then
   The possibility that they are the same Source is MEDIUM.
```

Figure 2: An Example of Fuzzy Inference Rules and Data-driven Neural Nets

```
If Report exists for a vessel class Rose to be in the vicinity, and
   Source likely to be associated with Rose has been detected,
TheExpect to find other Source types associated with Rose class.
```

Figure 3: An Example of Fuzzy Inference Rules

is activated by the arrival of data). From the symbolic system's point of view, neural net modules act as predicates in a fuzzy rule's condition side that check a "soft" (fuzzy) condition and return a number between zero and one indicating the degree of matching (e.g., the membership value of a fuzzy set). Figure 2 shows an example of fuzzy inference rule[1] where source refers to some noise-producing objects, such as propellers and shafts on ships. Fuzzy sets in the rules are expressed in uppercase. Suppose a neural net NN_1 classifies sensor data from hydrophones into possible sources of the noise. The fuzzy inference rule will combine the output of the neural net with other symbolic information (e.g., the reason a source was lost, the location of the sources) to determine the applicability of the rule.

In addition to use the output of a neural net in a data-driven fashion, a fuzzy inference rule can also invoke a neural net in a goal-driven fashion. For instance, the fuzzy inference rule in Figure 3 creates an expectation about the existence of certain source types. This expectation can be verified by several neural net modules that classifies noise sources associated with Rose class vessel.

[1]The examples in Figures 2 and 3 are two rules in HASP, a Blackboard system that analyzes sensor data from hydrophone arrays for ocean surveillance mission [8].

3.2 Modeling Meta-level Knowledge about Neural Networks

For a symbolic system to control neural nets and to use them as modules that extend its reasoning capabilities, it needs some information about the performance and the functional behaviors (e.g., input/output descriptions) of the neural nets. Such information is particularly crucial for integrating neural nets and symbolic systems, as they can not easily communicate with each other otherwise. Our approach is to symbolically represent information about classes of neural networks and individual neural networks, using a principled frame-based knowledge representation mechanism, called *term subsumption languages*[17]. Doing so offers three important advantages.

1. The model describes the functional behavior of neural networks in a way that helps the symbolic system invoke neural nets to extend its capabilities. For instance, an input/output description of a neural net allows the symbolic expert system to tell when a question it is working on can be answered by activating a particular neural net.

2. It provides the basic structure for our method inheritance mechanism (see Section 3.3). This allows general methods and specific methods to be described at their appropriate abstraction level, which facilitates the sharing of common methods and a saving of effort in developing and modifying them.

3. Finally, this approach enables the symbolic system to reason about the behavior of neural networks using automatic classification reasoning capabilities of term subsumption systems[18], which extend the system's knowledge about neural nets beyond what's stated explicitly in the model.

Figure 4 shows an example of meta-level knowledge that might be kept about a neural net for classifying the hostility of a submarine based on its location, speed, direction of movement, and depth. Several attributes need explanation. **Reliability** is the cumulative performance measure of the neural net, while **performance-measure** records the performance of the neural net's last activation. The **reliability-threshold** is the minimum reliability of the neural network that the system can tolerate. A neural net needs to be modified when its reliability is below its threshold value.

CLASP provides a rich term subsumption language, LOOM [13], for modeling meta-level knowledge about neural nets. *Term Subsumption Languages* are knowledge representation formalisms that employ a formal language, with a formal semantics, for the definition of terms (more commonly referred to as concept or classes), and that deduce whether one term subsumes (is more general that) another [17]. These formalisms generally descend from the ideas presented in KL-ONE [5]. Term subsumption languages

```
Name:  BACK1
Type:  Three-layer-feedforward
Learning:  Back-propagation
Input:  Location, speed, direction, depth
Output:  Hostility
Training-status:  Trained
Performance-measure:  Satisfactory
Reliability:  0.9
Reliability-threshold:  0.7
```

Figure 4: Meta-level Knowledge about a Neural Net

are a generalization of both semantic networks and frames because the languages have well-defined semantics, which is often missing from frames and semantic networks [20, 4]. The major benefit of using a term subsumption language (e.g., LOOM) to model the neural nets lies in its strong support for developing a consistent and coherent class taxonomy. This can be illustrated by the following example. Suppose the model defines that (1) a **possible-spurious-recognition-net** is any **noise-sensitive-net** which has two examplars that differ in less than two pixels; and, (2) CG_1 is a neural net module of type **Carpenter-Grossberg-net**, which is a kind of **noise-sensitive-net**. If CG_1 has two examplars that differ only in one pixel, LOOM will infer that CG_1 is a **possible-spurious-recognition-net**. Thus, using a term subsumption language to model the neural net taxonomy improves the consistency of the taxonomy, avoids redundancy in the model, and minimizes human errors introduced into the meta-level knowledge base.

3.3 Control Tasks and Methods

To link a symbolic system and neural net modules, a hybrid system needs to define a set of functions that interface between them. These functions facilitate the construction of a layered hybrid system by serving as the intermediate layer between the symbolic system and the neural nets. This layered approach means that hybrid systems will be built in a flexible and extensible way because we can extend the intermediate layer with minimum modification to the symbolic system and the neural nets.

Our approach to building the intermediate level has two major aspects. First, we use a set of generic functions (called *control tasks*) to define the interaction between the symbolic system and the neural networks. Second, we use methods in object-oriented

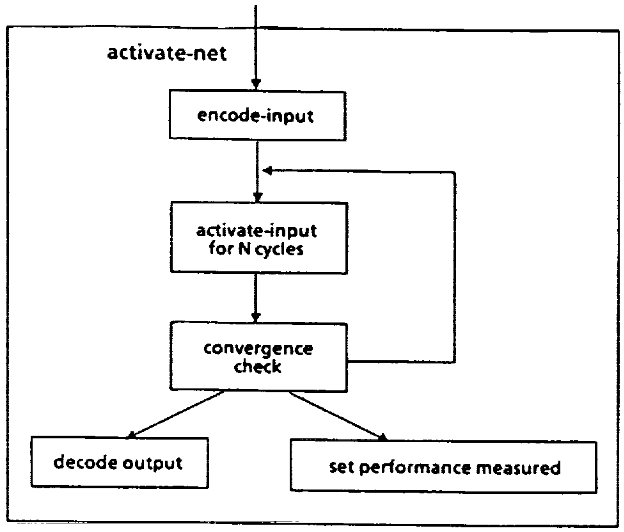

Figure 5: An example of control task decomposition

programming to implement control tasks.

Conceptually, we can view control tasks as messages sent back and forth between symbolic systems and neural networks. Symbolic systems use control tasks to activate and modify neural network modules; these, in turn, use control tasks to inform the symbolic system about their input/output behaviors. For example, the symbolic system would send an **activate-net** message to a neural network object in order to activate its corresponding neural network module[2]. Conversely, the neural network module would send a **set-performance-measure** message to the neural net object in order to update the neural net's **performance-measure** (possibly causing monitoring rules to be triggered). Some of the basic control tasks supported by the architecture may include: **activate-net**, **train-net**, **set-training-status**, **set-performance-measure**, **update-reliability**, and **remove-training-pattern**.

Our approach increases the reusability of modules and reduces the cost of developing and maintaining the system in two ways. First, it separates the purpose of a task from its implementation. Using control tasks to indicate "what needs to be done" allows the symbolic system and the neural nets to interact at an abstraction level that is independent of their detailed implementations. Second, our approach facilitates decomposing tasks into subtasks that can be shared by multiple neural nets. For example, the control task **activate-net** can be further decomposed into five subtasks as shown in Figure 5. By decomposing control tasks into subtasks, which are functional modules, we sep-

[2]A neural network module can also be activated by the arrival of sensor data

arate application-specific modules (such as **encode-input** and **decode-output**[3]) from application-independent modules (such as **activate-input**).

CLASP offers a mechanism for defining generic functions (also called *operators*) that can be invoked by rules or by function calls in any program [22]. CLASP's capability to invoke generic functions by rules and by procedural call is important because it allows the symbolic system to invoke control tasks by rule triggering, and the neural net modules to initiate control tasks through procedural invocations.

Control tasks will be implemented using method inheritance mechanisms in CLASP's object-oriented programming capabilities[4]. The **methods** implementing control tasks are attached to the neural net objects, which are organized into a taxonomy. An individual neural net inherit all its methods from its parents in the taxonomy. To implement a control task for a neural net N, the architecture finds a method for the task that is inherited from the most specific parent of N. This approach increases the reusability of methods, and avoids redundancy in defining similar methods. For example, although different bidirectional associative memories (BAM's) may differ in how they encode and decode symbolic information, they could all share the same **activate-input** method.

3.4 Fuzzy Action Rules

In addition to storing meta-level information about neural nets and specifying possible control actions on a neural net, the symbolic system needs a mechanism for recognizing situations within neural nets that indicate a need for action. Even though production systems in artificial intelligence offers such a capability, they do not address the issue of *partial matching* (accepting an approximate fit between observed data and a rule's condition). A production system that takes into account the degree of partial matching will enable the system to respond in a flexible way even in the face of incomplete or noisy data.

Our approach is to use fuzzy action rules, a generalization of production rules, to issue control task to neural net modules. A fuzzy action rule can use the degree its condition is satisfied to adjust its action[5]. Depending on the partial matching result, a fuzzy action rule may or may not be deemed applicable. For example, a rule may be viewed applicable

[3]In our terminology, *encoding* refers to transforming raw sensor data or symbolic information into neural net representations, and *decoding* refers to transforming neural net representations back into symbolic form.

[4]Actually, the method-dispatching mechanism in CLASP is more general than those in object-oriented programming languages (e.g., SMALLTALK-80) in that it allows programmers to describe more complex situations in which a method applies [21].

[5]The partial matching results of fuzzy productions can also be used for conflict resolution[3].

```
If neural net N is a kind of bidirectional associative memory, and
   its classification results are UNSATISFACTORY,
Then decrease its reliability SLIGHTLY.

If the reliability of a neural net is VERY LOW,
Then set a goal to diagnose and fix the neural net and initialize
     the priority of the goal to be proportional to the degree of matching.
```

Figure 6: Two Rules that Monitor the Performance of a Neural Net

only if the degree of matching is greater than a threshold value.

To illustrate how we use fuzzy action rules to control activation, training, and performance of neural nets, two monitoring rules (paraphrased into English) are shown in Figure 6. They monitor neural net modules by updating and acting on the modules' performance measures. The first rule illustrates how our neural net taxonomy allows rules to apply over whole classes of neural net modules. The second rule demonstrates that actions of rules can be high level tasks which cause the symbolic system to pursue further problem solving and diagnostic reasoning.

4 Summary

We have outlined a novel hybrid architecture that uses fuzzy logic to integrate neural networks and knowledge-based systems. Our approach offers important synergistic benefits to neural nets, approximate reasoning, and symbolic processing. Fuzzy inference rules extend symbolic systems with approximate reasoning capabilities, which are used for integrating and interpreting the outputs of neural networks. The symbolic system captures meta-level information about neural networks and defines its interaction with neural networks through a set of control tasks. Fuzzy action rules provides a robust mechanism for recognizing the situations about neural networks that require certain control actions. The neural nets, on the other hand, offers flexible classification and adaptive learning capabilities, which is crucial for dynamic and noisy environment. By combining neural nets and symbolic systems at their functional level through the use of fuzzy logic, our approach alleviates current difficulties in reconciling differences between the low-level data processing mechanisms of neural nets and AI systems.

Our technical approach to achieving this high-level integration also offers several advantages concerning the development and the maintenance of applications based on the hybrid architecture:

1. Fuzzy logic serves as a natural bridge that brings together subsymbolic processing of neural networks and symbolic reasoning in knowledge-based systems.

2. The interface between symbolic system and neural nets can be modified easily because it is implemented using a layered and modular approach.

3. Meta-level knowledge about neural nets is stored in a taxonomic structure that facilitates the sharing of information and procedures (e.g., methods).

4. Representing information about neural nets using a principled AI knowledge representation language enables the system to reason about the behavior of neural nets using AI deductive reasoning capabilities.

The hybrid architecture, which supports communication and control across multiple cooperative neural nets through the use of fuzzy rules, enables the construction of modular, flexible, and extensible intelligent systems, reduces the effort for developing and maintaining such systems, and facilitates their application to complex real world problems that need to perform low-level data classification as well as high-level problem solving in the presence of uncertainty and incomplete information.

Acknowledgements

I would like to thank Bart Kosko, Robert Neches, Paul Rosenbloom, John Granacki, and C. C. Lee for helpful discussions and comments about the paper. I am mostly in debt to Professor Lotfi A. Zadeh, from whom I receive constant encouragements and inspirations.

References

[1] John R. Anderson. *The Architecture of Cognition.* Harvard University Press, 1983.

[2] M. Arbib, E. Conklin, and J. Hill. *From Schema Theory to Language.* Oxford University Press, 1987.

[3] James Bowen and Jianchu Kang. Conflict resolution in fuzzy forward chaining production systems. In *Proceedings of AAAI-88*, pages 117–121, 1988.

[4] R. J. Brachman. What is-a is and isn't: An analysis of taxonomic links in semantic networks. *Computer*, 16(10):30–36, October 1983.

[5] R.J. Brachman and J.G. Schmolze. An overview of the kl-one knowledge representation system. *Cognitive Science*, pages 171–216, August 1985.

[6] Hon Wai Chun, Lawrence A. Bookman, , and Niki Afshartous. Network regions: Alternatives to the winner-take-all structure. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, pages 380–387. IJCAI, Morgan Kaufman, 1987.

[7] M. G. Dyer, M. Flowers, and Y. A. Wang. Weight mattrix = pattern of activation: Encoding semantic networks as distributed representations in dual, a pdp architecture. Technical Report UCLA-AI-88-5, Artificial Intelligence Lab., University of California at Los Angeles, 1988.

[8] Robert Engelmore and Tony Morgan. *Blackboard Systems*. Addison-Wesley Publishing, 1988.

[9] Stephen I. Gallant. Connectionist expert systems. *Communications of the ACM*, 31(2), February 1988.

[10] Bart Kosko. Fuzzy associative memories. In A. Kandel, editor, *Fuzzy Expert Systems*. Addison-Wesley, MA, 1987.

[11] Bart Kosko. Bidirectional associative memories. *IEEE Transactions on Systems, Mand, and Cybernetics*, 18(1), 1988.

[12] C. C. Lee. A self-learning rule-based controller employing fuzzy logic and neural net concepts. *International Journal of Intelligent Systems (to appear)*, 1990.

[13] R. M. MacGregor. A deductive pattern matcher. In *Proceedings of AAAI-88*, 1988.

[14] James L. McClelland and David E. Rumelhart. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 2. Bradford Books, Cambridge, MA, 1986.

[15] R. Neches, W. Swartout, , and J. Moore. Enhanced maintenance and explanation of expert systems through explicit models of their development. *Transactions On Software Engineering*, SE-11(11):1337–1351, November 1985.

[16] Yoh-Han Pao. *Adaptive Pattern Recognition and Neural Networks*. Addison-Wesley Publishing, 1989.

[17] Peter F. Patel-Schneider, Bernd Owsnicki-Klewe, Alfred Kobsa, Nicola Guarino, Robert MacGregor, William S. Mark, Deborah McGuinness, Bernhard Nebel, Albrecht Schmiedel, and John Yen. Report on the workshop on term subsumption languages in knowledge representation. to appear in AI Magazine, 1990.

[18] James Schmolze and Thomas Lipkis. Classification in the kl-one knowledge representation system. In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*. IJCAI, 1983.

[19] David S. Touretzky and Geoffrey E. Hinton. Symbols among the neurons: Details of a connectionist inference architecture. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, pages 238–243. IJCAI, Morgan Kaufman, 1985.

[20] William A. Woods. Whats's in a link: Foundations for semantic networks. In Daniel Bobrow and Allan Collins, editors, *Representation and Understanding: Studies in Cognitive Science*. Academic Press, 1975.

[21] John Yen. A principled approach to reasoning about specificity of rules. In *Proceedings of AAAI-90*, August 1990.

[22] John Yen, Hsiao-Lei Juang, and Robert MacGregor. Using object-oriented programming to enhance the maintainability of expert systems. in review, 1990.

[23] John Yen, Robert Neches, and Robert MacGregor. Using terminological models to enhance the rule-based paradigm. In *Proceedings of the Second International Symposium on Artificial Intelligence*, Monterrey, Mexico, October 25-27 1989.

[24] L. A. Zadeh. Possibility theory and soft data analysis. In L. Cobb and R. M. Thrall, editors, *Mathematical Frontiers of the Social and Policy Sciences*, pages 69–129. Westview Press, Boulder, Colorado, 1981.

AUTHOR INDEX

Anderson, James A. 109

Berenji, Hamid R. 1
Bezdek, James C. 145
Bishop, Thomas 115

Collins, Dean R. 109

Gallant, A. R. 61
Gately, Michael T. 109
Greenwood, Dan 115

Hayashi, Isao 171
Hirota, Kaoru 185
Hornik, K 61

Ikoma, Norikazu 185

Jani, Yashvant 81

Keller, James M. 205
Kosko, Bert 3

Lea, Robert N. 81
Lee, Chuen-Chien 197

Nomura, Hiroyoshi 171

Pal, Sankar K. 213
Penz, P. Andrew 109

Ruspini, Enrique H. 235

Shelton, Robert O. 63
Shew, Kenneth 115
Stevenson, Fareed 115
Stinchcombe, M. 61
Symon, James R. 161

Taber, Rod 31
Teh, H. H. 97

Villarreal, James A. 63

Wang, P. Z. 97
Wakami, Noboru 171
Watanabe, Hiroyuki 161
Werbos, Paul J. 153
White, Halbert 61
Wu, Z. Q. 97

Yen, John 221

PRECEDING PAGE BLANK NOT FILMED