EXPERIMENTS ON NEURAL NETWORK ARCHITECTURES FOR FUZZY LOGIC

James M. Keller
Electrical and Computer Engineering
University of Missouri - Columbia
Columbia, MO 65211

ABSTRACT

The use of fuzzy logic to model and manage uncertainty in a rule-based system places high computational demands on an inference engine. In an earlier paper, we introduced a trainable neural network structure for fuzzy logic. These networks can learn and extrapolate complex relationships between possibility distributions for the antecedents and consequents in the rules. In this paper, the power of these networks are further explored. The insensitivity of the output to noisy input distributions (which are likely if the clauses are generated from real data) is demonstrated as well as the ability of the networks to internalize multiple conjunctive clause and disjunctive clause rules. Since different rules (with same variables) can be encoded in a single network, this approach to fuzzy logic inference provides a natural mechanism for rule conflict resolution.

1. INTRODUCTION.

In dealing with automated decision making problems, and computer vision in particular, there is a growing need for modeling and managing uncertainty. Computer vision is beset with uncertainty of all types. A partial list of the causes of such uncertainty include:

complexity of the problems,
questions which are ill-posed,
vagueness of class definitions,
imprecisions in computations,
noise of various sorts,
ambiguity of representations, and
problems in scene interpretation.

Rule-based approaches for handling these problems have gained popularity in recent years [1-6]. They offer a degree of flexibility not found in traditional approaches. The systems based on classical (crisp) logic need to incorporate, as an add-on, the processing of the uncertainty in the information. Methods to accomplish this include heuristic approaches [7, 8], probability theory [9,10], Dempster-Shafer belief theory [4,5,11], and fuzzy set theory [5,6,12-14].

Fuzzy logic, on the other hand, is a natural mechanism for propagating uncertainty explicitly in a rule base. All propositions are modeled by possibility distributions over appropriate domains. For example, a computer vision system may have rules like

IF the range is LONG, THEN
    the prescreener window size is SMALL;

or

IF the color is MOSTLY RED, THEN
    the steak is MEDIUM RARE is TRUE.

Here, LONG, SMALL, MOSTLY RED and TRUE are modeled by fuzzy subsets over appropriate domains of discourse. The possibility distributions can be generated from various histograms of feature data extracted from images, fuzzification of values produced by pattern recognition algorithms, experts expressing (free form) opinions on some questions, or possibly generated by a neural network learning algorithm.

The generality inherent in fuzzy logic comes at a price. Since all operations involve sets, rather than numbers, the amount of calculations per inference rises dramatically. Also, in a fuzzy logic system, generally more rules can be fired at any given instant. One approach to combat this computational load has been the development of special purpose chips which perform particular versions of fuzzy inference [15]. Artificial Neural Networks offer the potential of parallel computation with high flexibility. In an earlier paper [16], we introduced a backpropagation neural network structure to implement fuzzy logic inference. In this paper we demonstrate further properties of that network. In particular, we show the insensitivity of the networks to noisy input distributions and to their ability to internalize rules with multiple conjunctive and disjunctive antecedent clauses.

2. FUZZY LOGIC AND NEURAL NETWORKS.

The original fuzzy inference mechanism extended the traditional modus ponens rule which states that from the propositions

P_1: **If X is A Then Y is B**

and P_2: **X is A,**

we can deduce **Y is B**. If proposition P_2 did not exactly match the antecedent of P_1, for example, **X is A'**, then the modus ponens rule would not apply. However, in [17], Zadeh extended this rule if A, B, and A' are modeled by fuzzy sets, as suggested above. In this case, P_1 is characterized by a possibility distribution:

$$\Pi_{(X|Y)} = R \text{ where}$$

$$\mu_R(u,v) = \max\{(1-\mu_A(u)),\ \mu_B(v)\}.$$

It should be noted that this formula corresponds to the statement "not A or B", the logical translation of P_1. An alternate translation of the rule P_1 which corresponds more closely to multivalue logic is

$$\mu_R(u,v) = \min\{1,\ \{(1 - \mu_A(u)) + \mu_B(v)\}\},[17],$$

called the bounded sum.

In either case, Zadeh now makes the inference **Y is B'** from μ_R and $\mu_{A'}$ by

$$\mu_{B'}(v) = \max_u\{\min\{\mu_R(u,v),\mu_{A'}(u)\}\}.$$

This is called the compositional rule of inference.

While this formulation of fuzzy inference directly extends modus ponens, it suffers from some problems [18,19]. In fact, if proposition $\mathbf{P_2}$ is **X is A**, the resultant fuzzy set is not exactly the fuzzy set B. Several authors [18-20] have performed theoretical investigations into alternative formulations of fuzzy implications in an attempt to produce more intuitive results.

In using fuzzy logic in real rule-based systems, the possibility distributions for the various clauses in the rule base are normally sampled at a fixed number of values over their respective domains of discourse, creating a vector representation for the possibility distribution. Table I shows the sampled versions of the "trapezoidal" possibility distributions, used in the simulation study, sampled at integer values over the domain [1,11]. Clearly , the sampling frequency has a direct effect on the faithfulness of the representation of the linguistic terms under consideration and also on the amount of calculation necessary to perform inference using a composition rule. For a single antecedent clause rule, the translation becomes a two dimensional matrix and the inference is equivalent to maxtrix-vector multiplication. As the number of antecedent clauses increases, the storage (multidimensional matrices) and the computation in the inference process grows exponentially.

Neural network structures offer a means of performing these computations in parallel with a compact representation. But the ability of such a network to generalize from an existing training set is the most valuable feature. In [16], we introduced the neural network architecture for fuzzy logic. Figure 1 displays a three layer feed-forward neural network which is used in fuzzy logic inference for conjunctive clause rules. It consisted of an input layer to receive the possibility distributions of the antecedent clauses, one hidden layer to internalize a representation of the relationships, and an output layer to produce the possibility distributions of the consequent.

The input layer is not fully connected to the hidden layer. Instead, each antecedent clause has its own set of hidden neurons to learn the desired relationship. This partitioning of the hidden layer was done to ease the training burden for multiple clause rules, and to treat each input clause with its hidden units as a functional block. The training was performed using the standard back propagation technique [21].

3. EXPERIMENTS.

The neural network architecture performed very well in generalizing the complex relationships between inputs and outputs. Table II (from [16]) shows the results of the training and testing of a network to implement the rule: IF X is LOW Then Y is HIGH; whereas Table III gives the situation for a rule with two conjunctive antecedent clauses. In both cases, the performance of the networks matched our intuitive expectation.

Figure 2 shows typical responses of a neural network to noise in the input clause. It can be seen that the errors in the result are of the same order as the error in the input . If the networks are trained with fewer relationships, e.g. the traditional modus ponens expectations, this error drops significantly.

In order to implement rules with disjunctive antecedent clauses, networks with two hidden layers were necessary. Table IV displays training relationships for a two clause disjunctive rule. Note that there are 23 input/output triples necessary to enable the network to respond appropriately. The training, using backpropagation, of a single hidden layer network, of the type shown in figure 1, failed to converge on this complex training set. This caused us to investigate a two hidden layer structure where the first hidden layer was the same as in figure 1 and the second hidden layer contained 6 neurons totally connected to those of the first hidden layer and to the nodes of the output layer. This network converged in 4073 passes through the training set with a total-sum-of-squared error of less than 0.001 for the entire training ensemble. We feel that this is a remarkable achievement, given the diversity of the responses to the antecedent possibility distributions which were necessary.

This disjunctive structure was further tested with 18 input pairs of clauses including twelve pairs with varying amounts of additive gaussian noise. For this test set the average total-sum-of-squared-error per trial was 0.075. In other words, the match to the expected output in all cases was very good.

As a final note, in [16] we demonstrated that a neural network structure of this type could encode multiple different rules which shared common antecedent clause variables. The packing of several rules into a single network has a surprising side benefit of providing a natural means of conflict resolution in fuzzy logic.

4. CONCLUSION.

Fuzzy logic is a powerful tool for managing uncertainty in rule-based systems. Neural network architectures offer a means of relieving some of the computational burden inherent in fuzzy logic. Also, these structures can be trained to learn and extrapolate complex relationships between antecedents and consequents, they are relatively insensitive to noise in the inputs, and provide a natural mechanism for conflict resolution.

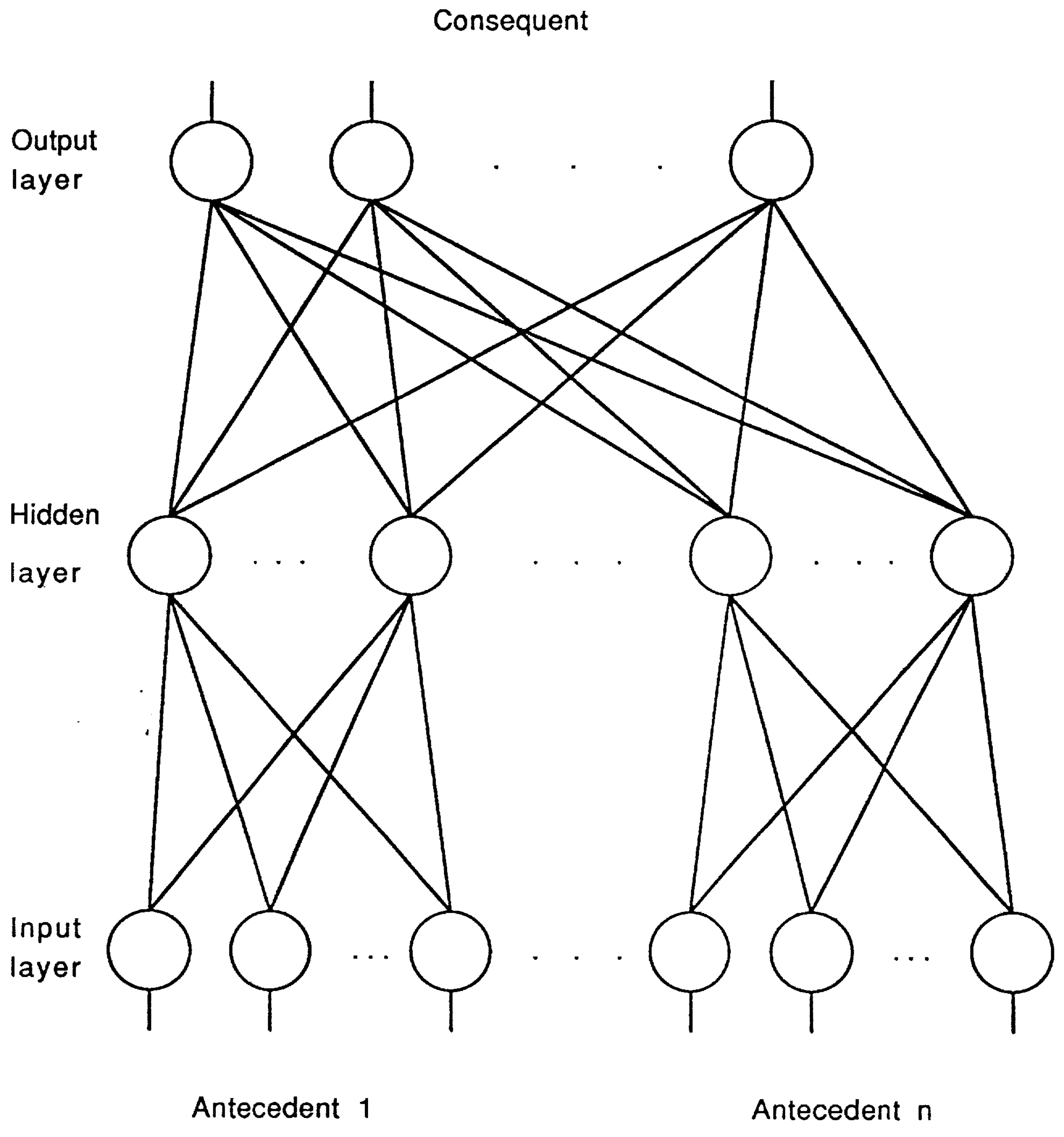
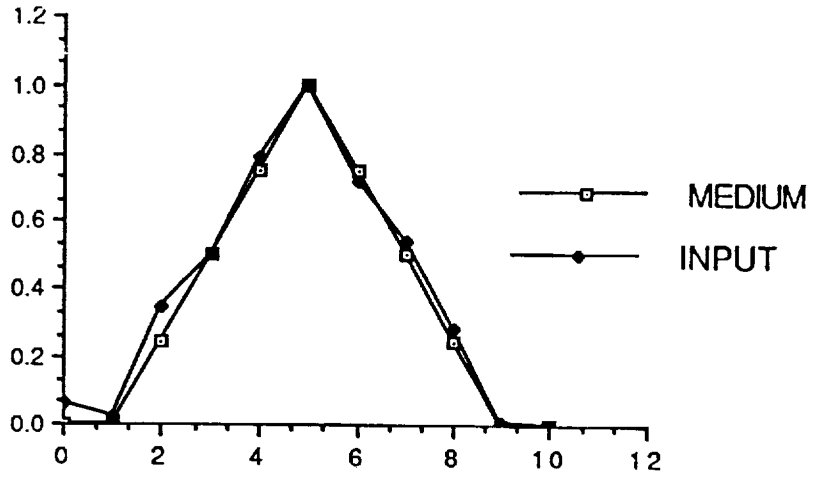


Figure 1. A three layer feed forward neural network for fuzzy logic inference

Rule: IF X is MEDIUM THEN Y is HIGH

MEDIUM	.00	.00	.25	.50	.75	1.0	.75	.50	.25	.00	.00
INPUT	.06	.02	.35	.50	.79	1.0	.72	.54	.29	.01	.00

TSS error = 0.020



HIGH	.00	.00	.00	.00	.00	.00	.20	.40	.60	.80	1.0
OUTPUT	.00	.00	.00	.00	.00	.00	.28	.48	.67	.84	1.0

TSS error = 0.019

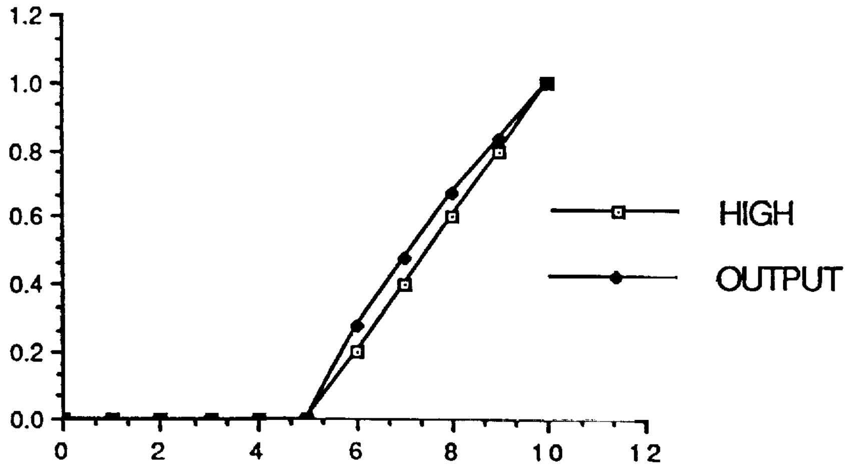
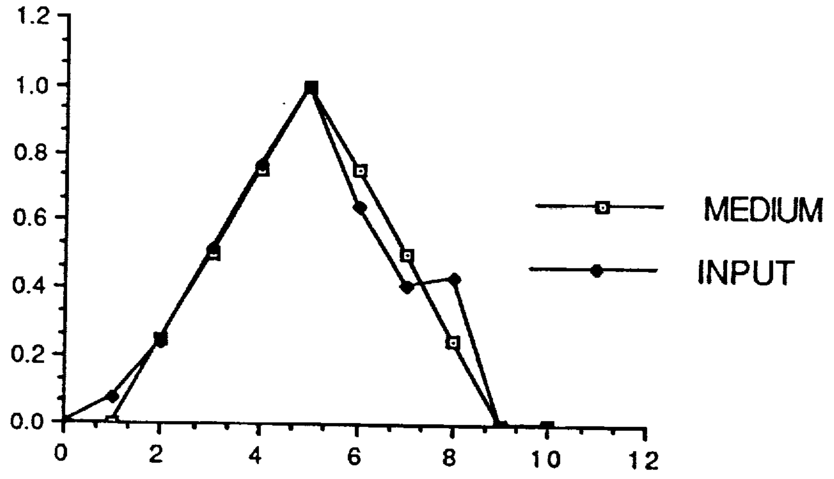


Figure 2(a) Response of rule network to an input with small amount of additive gaussian noise.

MEDIUM	**.00**	**.00**	**.25**	**.50**	**.75**	**1.0**	**.75**	**.50**	**.25**	**.00**	**.00**
INPUT	**.00**	**.08**	**.24**	**.52**	**.77**	**1.0**	**.64**	**.41**	**.43**	**.00**	**.00**

TSS error = 0.060



HIGH	**.00**	**.00**	**.00**	**.00**	**.00**	**.00**	**.20**	**.40**	**.60**	**.80**	**1.0**
OUTPUT	**.00**	**.00**	**.00**	**.00**	**.00**	**.00**	**.25**	**.46**	**.65**	**.83**	**1.0**

TSS error = 0.010

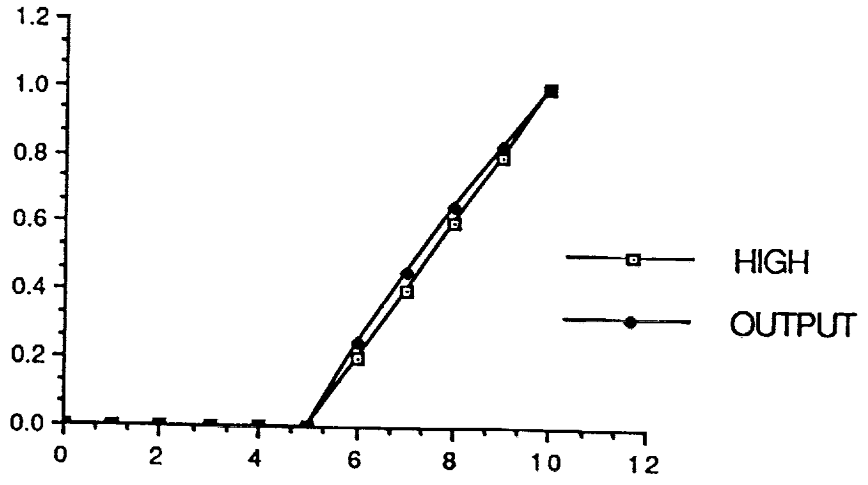


Figure 2(b) Response of rule network to an input with a larger amount of additive gaussian noise.

Table I. The meaning of linguistic terms defined on the domain [1,11] and sampled at integer points.

Label	Membership										
LOW	1.00	0.67	0.33	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
VERY LOW	1.00	0.45	0.11	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MORL LOW	1.00	0.82	0.57	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
NOT LOW	0.00	0.33	0.67	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
NOISY LOW (1)	1.00	0.70	0.40	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
NOISY LOW (2)	1.00	0.70	0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
NOISY MEDIUM	0.00	0.00	0.30	0.53	0.81	1.00	0.80	0.50	0.20	0.00	0.00
SHIFTED LOW	1.00	1.00	1.00	0.67	0.33	0.00	0.00	0.00	0.00	0.00	0.00
MEDIUM	0.00	0.00	0.25	0.50	0.75	1.00	0.75	0.50	0.25	0.00	0.00
MORL MEDIUM	0.00	0.00	0.50	0.71	0.87	1.00	0.87	0.71	0.50	0.00	0.00
NOT MEDIUM	1.00	1.00	0.75	0.50	0.25	0.00	0.25	0.50	0.75	1.00	1.00
HIGH	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.40	0.60	0.80	1.00
VERY HIGH	0.00	0.00	0.00	0.00	0.00	0.00	0.04	0.16	0.36	0.64	1.00
MORL HIGH	0.00	0.00	0.00	0.00	0.00	0.00	0.45	0.63	0.77	0.89	1.00
UNKNOWN	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

MORL = more or less.

Note: Very^{n}A is determined by $\mu_{\mathrm{Very}^n A}(x) = \mu_A(x)^{n+1}$

MORL^{n}A is determined by $\mu_{\mathrm{MORL}^n A}(x) = [\mu_A(x)]^{1/n+1}$

Table II. Performance of Fuzzy Logic Rule network with 8 hidden neurons for rule **IF X is LOW THEN Y is HIGH.**

A. Training Data*

Input	Output
LOW	HIGH
VERY LOW	VERY HIGH
MORL LOW	MORL HIGH
NOT LOW	UNKNOWN

* Training terminated when the total sum of squared error dropped below ϵ = .001

B. Testing Results

Input	Expected Output	Actual Output											Total Sum Squared Error
VERY^2 LOW	VERY^2 HIGH	.00	.00	.00	.00	.00	.00	.03	.10	.27	.56	1.0	.007
MORL^2 LOW	MORL^2 HIGH	.00	.01	.01	.01	.00	.01	.56	.71	.82	.91	1.0	.030
MEDIUM	UNKNOWN	.99	.99	.99	.99	.99	.99	.99	.99	.99	.99	1.0	.001
VERY MEDIUM	UNKNOWN	.98	.98	.98	.98	.98	.98	.99	.99	.99	.99	1.0	.003
MORL MEDIUM	UNKNOWN	.99	.99	.99	.99	.99	.99	.99	.99	.99	.99	.99	.001
HIGH	UNKNOWN	.99	.99	.99	.99	.99	.99	.99	.99	.99	.99	.99	.001
NOISY LOW (1)	HIGH	.00	.00	.00	.00	.00	.00	.26	.47	.66	.83	1.0	.013
NOISY LOW (2)	HIGH	.00	.00	.00	.00	.00	.00	.19	.39	.59	.80	1.0	.0001
SHIFTED LOW	———	.09	.09	.12	.09	.09	.09	.91	.92	.94	.97	1.0	———

Table III. Performance of a two antecedent clause Fuzzy Logic Rule network with 16 hidden neurons (two groups of eight).

A. Training Data*

Input	Output
(LOW,MEDIUM)	HIGH
(VERY LOW,VERY MEDIUM)	VERY HIGH
(MORL LOW,MORL MEDIUM)	MORL HIGH
(NOT LOW,MEDIUM)	UNKNOWN
(LOW,NOT MEDIUM)	UNKNOWN

* Training converged in 1823 iterations.

B. Testing Results

Input	Actual Output											Closest Linguistic Term
(NOISY LOW(1),MEDIUM)	.00	.00	.00	.00	.00	.00	.20	.40	.60	.80	1.0	HIGH
(NOISY LOW(2),MEDIUM)	.00	.00	.00	.00	.00	.00	.19	.40	.60	.80	1.0	HIGH
(VERY^2 LOW,MEDIUM)	.00	.00	.00	.00	.00	.00	.19	.38	.60	.80	1.0	HIGH
(NOISY LOW(1),NOISY MEDIUM	.00	.00	.00	.00	.00	.00	.20	.41	.61	.81	1.0	HIGH
(LOW,VERY^2 MEDIUM)	.00	.00	.00	.00	.00	.00	.05	.17	.36	.64	1.0	VERY HIGH
(VERY^2 LOW,VERY^2 MEDIUM)	.01	.01	.01	.01	.01	.01	.03	.12	.29	.58	1.0	VERY^2 HIGH
(MORL^2 LOW,MORL^2 MEDIUM)	.01	.01	.01	.01	.01	.01	.55	.70	.81	.91	1.0	MORL^2 HIGH
(NOT LOW,NOT MEDIUM)	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	UNKNOWN
(LOW,SHIFTED MEDIUM)	.97	.97	.97	.97	.97	.97	.99	.99	.99	1.0	1.0	UNKNOWN
(MEDIUM,LOW)	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	UNKNOWN

Table IV. Training Data for the two disjunctive clause rule:
IF X is LOW <u>OR</u> Y is MEDIUM THEN Z is HIGH.

Input	Output
(Very, MorL) LOW: *	(Very, MorL) HIGH
*; (Very, MorL) MEDIUM	(Very, MorL) HIGH
Not LOW; Not MEDIUM	UNKNOWN
MEDIUM; LOW	UNKNOWN
HIGH; LOW	UNKNOWN
HIGH; Very LOW	UNKNOWN
UNKNOWN, HIGH	UNKNOWN

* = LOW, MEDIUM, HIGH

Training converged in 4073 iterations, with TSS error for entire training set less than 0.001

5. REFERENCES.

1. Fikes, R., and Nilsson, N., "STRIPS: a new approach to the application of theorem proving to problem solving", Artificial Intelligence, 2, 3/4, 1971, pp. 189-208.

2. Barrow, H. and Tenenbaum, J. "MSYS: a system for reasoning about scenes", Technical Note 121, AI Center, SRI International, March 1976.

3. Brooks, R., Greiner, R., and Binford, T., "Progress report on a model-based vision system", Proc. Image Understanding Workshop, L. Baumann, ed., 1978, pp. 145-151.

4. Riseman, E., and Hanson, A, "A methodology for the development of general knowledge-based version systems", in Vision, Brain, and Cooperative Computation, M. Arbib and A. Hanson, Eds., MIT Press, Cambridge, MA, 1988, pp. 285-328.

5. Wootton, J. Keller, J., Carpenter, C., and Hobson, G., "A multiple hypothesis rule-based automatic target recognizer," in Pattern Recognition, Lecture Notes in Computer Science, Vol. 301, Kittler, J. ed., Springer-Verlag, Berlin, 315-324, 1988.

6. Nafarieh, A., and Keller, J. "A fuzzy logic rule-based automatic target recognizer", Int. J. Intell. Systems, accepted for publication, 1990.

7. Shortliffe, E., and Buchanan, "A model of inexact reasoning in medicine", Math Biosci, 23, 1975, pp. 351-379.

8. Cohen, P., Heuristic Reasoning About Uncertainty: An Artificial Intelligence Approach. Pitman Advanced Publishing Program, 1985.

9. Pearl, J., "Fusion propagation and structuring in belief networks", Art. Intell., 29, no. 3, 1986, pp. 241-288.

10. Cheeseman, P. "A method of computing generalized bayesian probability values for expert systems", Proc. Eight Int. J. Conf. on AI, Karlsrulie, West Germany.

11. Li, Z. "Uncertainty management is a pyramid vision system", Int. J. Approx. Reasoning, 3, no. 1, 1989, pp. 59-85.

12. Bonissone, P. and Tong, R. "Editorial: Reasoning with uncertainty in expert systems," Int. J. Man-Machine Studies, Vol. 22, 241-250, 1985.

13. Zadeh, L. "Fuzzy logic and approximate reasoning," Syntheses, Vol. 30, 407-428, 1975.

14. Keller, J., Hobson, G., Wootton, J. Nafarieh, A., and Leutkemeyer, K., "Fuzzy confidence measures in midlevel vision", IEEE T. Syst., Man, Cybern, 17, no. 4, 1987, pp. 676-683.

15. Togai, M. and Watanabe, H., "Expert system on a chip: an engine for real-time approximate reasoning", IEEE Expert, Fall 1986, pp. 55-62.

16. Keller, J. and Tahani, H. "Backpropagation neural networks for fuzzy logic", Info. Sciences, accepted for publication, 1990.

17. Zadeh, L. "The concept of a linguistic variable and its application to approximate reasoning," Information Sciences, Part 1, Vol. 8, pp. 199-249; Part 2, Vol. 8, pp. 301-357; Part 3, Vol. 9, pp. 43-80, 1975.

18. Nafarieh, A., "A new approach to inference in approximate reasoning and its application to computer vision," Ph.D. Dissertation, University of Missouri-Columbia, 1988.

19. Mizumoto, M., Fukami, S., and Tanaka, K., "Some methods of fuzzy reasoning," in Advances in Fuzzy Set Theory and Applications, Gupta, M., Ragade, R., and Yager, R., eds., North-Holland, Amsterdam, 1979, pp. 117-126.

20. Baldwin, J. and Guild, N. "Feasible algorithms for approximate reasoning using fuzzy logic," Fuzzy Sets and Systems, Vol. 3, 1980, pp. 225-251.

21. Rumelhart, D., McClelland, J., and the PDP Research Group, Parallel Distributed Processing, Vol. 1, MIT Press, Cambridge, MA, 1986.