The Third AFSS(1998), 707－712

Handwritten Digit Recognition with Softcomputing Techniques

Sung-Bae Cho

Dept. of Computer Science, Yonsei University
134 Shinchon-dong, Sudaemoon-ku, Seoul 120-749, Korea
Tel: +82-2-361-2720 Fax: +82-2-365-2579 E-mail: sbcho@csai.yonsei.ac.kr

Abstract

This paper presents several softcomputing techniques such as neural networks, fuzzy logic and genetic algorithms: Neural networks as brain metaphor provide fundamental structure, fuzzy logic gives a possibility to utilize top-down knowledge from designer, and genetic algorithms as evolution metaphor determine several system parameters with the process of bottom-up development. With these techniques, we develop a pattern recognizer which consists of multiple neural networks aggregated by fuzzy integral in which genetic algorithms determine the fuzzy density values. The experimental results with the problem of recognizing totally unconstrained handwritten numerals show that the performance of the proposed method is superior to that of conventional methods.

1 Introduction

Intelligent systems adaptively estimate continuous functions from data without specifying mathematically how outputs depend on inputs. For achieving some aspect of intelligent systems, softcomputing techniques such as neural networks, fuzzy logic and genetic algorithms have been proposed. However, their compensatory properties have prompted many researchers to combine them to produce more powerful systems [1, 8]. In this paper we present a framework for combining them, and develop a pattern recognition system as an implementation of the framework.

Neural networks and fuzzy logic estimate input-output functions. Unlike statistical estimators, they estimate a function without a mathematical model of how outputs depend on inputs. They learn from experience with numerical and sometimes linguistic sample data. Neural networks consist of numerous simple processing units and process information in a parallel distributed manner. They have several characteristics such as nonlinear mapping, parallel processing, learning and self-organization.

Fuzzy logic is characterized as extension of binary crisp logic. Each fuzzy rule has an antecedent part containing several preconditions, and a consequent part which prescribes the value. Fuzzy set is a class in which transition from membership to non-membership is gradual rather than abrupt, so that it allows partial membership. Fuzzy systems based on fuzzy logic directly encode structured knowledge but in a numerical framework.

Genetic algorithms are used as a search technique based on the mechanisms of natural selection and natural genetics. It combines survival of the fittest among string structures with a structured but randomized information exchange to form a search algorithm with some of the innovative flair of human problem solving. They efficiently exploit historical information to speculate on new search points with expected improved performance.

As described above, each technique has its own merits and demerits. To produce more powerful system, several integration and synthesis approaches have been taken. It is natural to think of a framework for high-performance system based on them. Neural networks can be used as a baseline system, because they are well recognized as a powerful input-output mapper. One of the weakest points, however, is that human operators cannot easily provide with any knowledge about the problem at hand. In this case, fuzzy logic is useful.

Fuzzy logic gives a possibility to utilize top-down knowledge from designer. Human operators are able to give their knowledge in neural networks by means of fuzzy membership functions. The membership functions are modified through learning process as fine tuning. On the other hand, genetic algorithms are a powerful tool for structure optimization of fuzzy logic and neural networks which provide evaluation functions for genetic algorithms. Figure 1 shows a schematic diagram of the general framework based on the hybridization of them.

To give an idea of how such a framework works out, this paper presents a pattern recognition system as an implementation. The system consists of a set of neural networks combined by fuzzy logic, especially Choquet integral. These methods consider the difference of performance of each network in the combining process. To determine the fuzzy density values required to design the Choquet integral, genetic algorithms are utilized for achieving the optimal solution to combine neural networks.

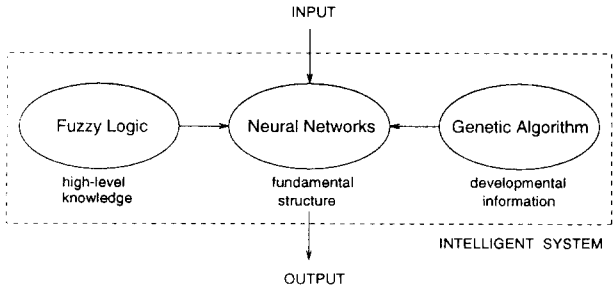

Figure 1: Schematic diagram of the softcomputing framework.

2 Multiple Neural Networks

When trying to solve a real-world problem by neural networks, we are faced with a large variety of learning algorithms and a vast selection of possible network architectures. After all the training, we choose the best network with a winner-take-all cross-validatory model selection. Recent theoretical and experimental studies, however, indicate that we can improve performance by considering methods for combining neural networks [2, 5]. One of the key issues of this approach is how to combine the results of the various networks to give the best estimate of the optimal result. If we have networks of different accuracy, however, it is obviously not good to take their simple average or simple voting.

Neural networks can be considered as a mapping device between an input set and an output set. Mathematically, a neural network represents a function F that maps I into O; $F : I \rightarrow O$, or $y = f(x)$ where $y \in O$ and $x \in I$. Since the classification problem is a mapping from the feature space to some set of output classes, we can formalize the neural network, especially two-layer feedforward neural network trained with the generalized delta rule, as a classifier.

Suppose a two-layer neural network classifier with T neurons in the input layer, H neurons in the hidden layer, and c neurons in the output layer. Here, T is the number of features, c is the number of classes, and H is an appropriately selected number. The network is fully connected between adjacent layers. The operation of this network can be thought of as a nonlinear decision-making process: Given an unknown input $X = (x_1, x_2, \ldots, x_T)$ and the class set $\Omega = \{\omega_1, \omega_2, \ldots, \omega_c\}$, each output neuron produces y_i of belonging to this class by

$$P(\omega_i|X) \approx y_i = f\left\{\sum_{k=1}^{H} w_{ik}^{om} f\left(\sum_{j=1}^{T} w_{kj}^{mi} x_j\right)\right\}, \quad (1)$$

where w_{kj}^{mi} is a weight between the jth input neuron and the kth hidden neuron, w_{ik}^{om} is a weight from the kth hidden neuron to the ith class output, and f is a sigmoid function such as $f(x) = 1/(1 + e^{-x})$. The neuron having the maximum value is selected as the corresponding class.

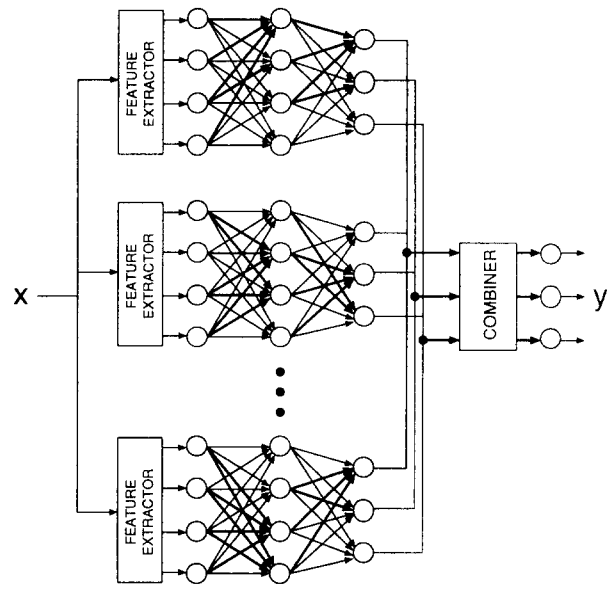

Figure 2: Multiple neural networks combined.

The network presented above trains on a set of example patterns and discovers relationships that distinguish the patterns. A network of a finite size, however, does not often load a particular mapping completely or it generalizes poorly. Increasing the size and number of hidden layers most often does not lead to any improvements. Furthermore, in complex problems such as character recognition, both the number of available features and the number of classes are large. The features are neither statistically independent nor unimodally distributed. The basic idea of the multiple network scheme is to develop n independently trained neural networks with relevant features, and to classify a given input pattern by utilizing combination methods to decide the collective classification [2] (Figure 2). Then it naturally raises the question of obtaining a consensus on the results of each individual network or expert.

3 Fuzzy Integrals

In this section, we describe the method that utilizes the fuzzy integral for combining the neural networks. This method might produce better classification results with the subjectively assigned importances of individual networks.

The fuzzy integral is a nonlinear functional that is defined with respect to a fuzzy measure, especially g_λ-fuzzy measure introduced by Sugeno [4]. The ability

of the fuzzy integral to combine the results of multiple sources of information has been established in several previous works [7].

From the definition of a fuzzy measure g, Sugeno introduced the so-called g_λ-fuzzy measures satisfying the following additional property: For all $A, B \subset X$ and $A \cap B = \emptyset$,

$$g(A \cup B) = g(A) + g(B) + \lambda g(A)g(B), \quad \text{for some } \lambda > -1. \tag{2}$$

It affords that the measure of the union of two disjoint subsets can be directly computed from the component measures.

Using the notion of fuzzy measures, Sugeno developed the concept of the fuzzy integral, which is a nonlinear functional that is defined with respect to a fuzzy measure, especially g_λ-fuzzy measure [4, 7].

Definition 1: Let X be a finite set, and $h : X \to [0, 1]$ be a fuzzy subset of X. The fuzzy integral over X of the function h with respect to a fuzzy measure g is defined by

$$\begin{aligned} h(x) \circ g(\cdot) &= \max_{E \subseteq X} \left[\min \left(\min_{x \in E} h(x), g(E) \right) \right] \\ &= \sup_{\alpha \in [0,1]} \left[\min \left(\alpha, g(h_\alpha) \right) \right] \end{aligned} \tag{3}$$

where h_α is the α level set of h,

$$h_\alpha = \{ x \mid h(x) \geq \alpha \}. \tag{4}$$

The fuzzy integral of equation (3) is called the Sugeno integral. To get some intuition for the Sugeno integral we consider the following interpretation. $h(y)$ measures the degree to which the concept h is satisfied by y. The term $\min_{y \in E} h(y)$ measures the degree to which the concept h is satisfied by all the elements in E. Moreover, the value $g(E)$ is a measure of the degree to which the subset of objects E satisfies the concept measured by g. Then, the value obtained from comparing these two quantities in terms of the min operator indicates the degree to which E satisfies both the criteria of the measure g and $\min_{y \in E} h(y)$. Finally, the max operation takes the biggest of these terms. One can interpret the fuzzy integral as finding the maximal grade of agreement between the objective evidence and expectation.

The calculation of the fuzzy integral when Y is a finite set is easily given. Let $Y = \{y_1, y_2, \ldots, y_n\}$ be a finite set and let $h : Y \to [0, 1]$ be a function. Suppose $h(y_1) \geq h(y_2) \geq \ldots \geq h(y_n)$, (if not, Y is rearranged so that this relation holds). Then a fuzzy integral, e, with respect to a fuzzy measure g over Y can be computed by

$$e = \max_{i=1}^{n} \left[\min \left(h(y_i), \; g(A_i) \right) \right] \tag{5}$$

where $A_i = \{y_1, y_2, \ldots, y_i\}$.

Note that when g is a g_λ-fuzzy measure, the values of $g(A_i)$ can be determined recursively as

$$\begin{aligned} g(A_1) &= g(\{y_1\}) = g^1 \\ g(A_i) &= g^i + g(A_{i-1}) + \lambda g^i g(A_{i-1}), \quad \text{for } 1 < i \leq n. \end{aligned} \tag{6}$$

λ is given by solving the equation

$$\lambda + 1 = \prod_{i=1}^{n} (1 + \lambda g^i) \tag{7}$$

where $\lambda \in (-1, +\infty)$, and $\lambda \neq 0$. This can be easily calculated by solving an $(n-1)$st degree polynomial and finding the unique root greater than -1.

A quite different definition was proposed by Murofushi and Sugeno [4], using a function defined by Choquet in capacity theory as Choquet integral:

$$e = \sum_{i=1}^{n} (h(y_i) - h(y_{i-1})) g(A_i) \tag{8}$$

where $h(y_0) = 0$. Thus the calculation of the fuzzy integral with respect to a g_λ-fuzzy measure would only require the knowledge of the density function, where ith density, g^i, is interpreted as the degree of importance of the source y_i towards the final evaluation.

Let $\Omega = \{\omega_1, \omega_2, \ldots, \omega_c\}$ be a set of classes of interest. Note that each ω_i may, in fact, be a set of classes by itself. Let $Y = \{y_1, y_2, \ldots, y_n\}$ be a set of neural networks, and A be the object under consideration for recognition. Let $h_k : Y \to [0, 1]$ be the partial evaluation of the object A for class ω_k, that is, $h_k(y_i)$ is an indication of how certain we are in the classification of object A to be in class ω_k using the network y_i, where a 1 indicates absolute certainty that the object A is really in class ω_k and 0 implies absolute certainty that the object A is not in ω_k.

Corresponding to each y_i the degree of importance, g^i, of how important y_i is in the recognition of the class ω_k must be given. These densities can be subjectively assigned by an expert, or can be induced from data set. The g^i's define the fuzzy density mapping, and λ and g's are calculated. Then, using (8) the fuzzy integral can be calculated. Finally, the class ω_k with the largest integral value is chosen as the output class. Next section, we will describe how to determine the g's effectively by using genetic algorithms.

4 Genetic Algorithms

Evolution is a remarkable problem-solving machine [6]. First proposed by John Holland in 1975, GAs as one of computational implementations are an attractive class of computational models that mimic natural evolution to solve problems in a wide variety of domains. A genetic algorithm emulates biological evolutionary theories to solve optimization problems.

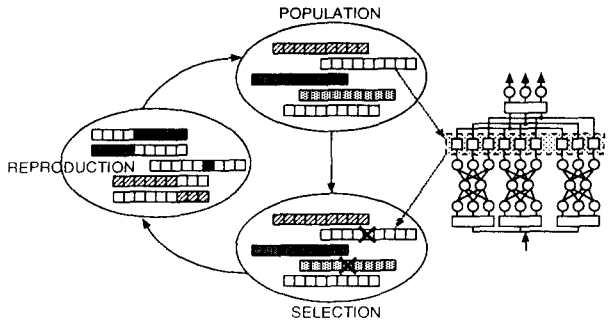

Figure 3: Overall procedure of the proposed method.

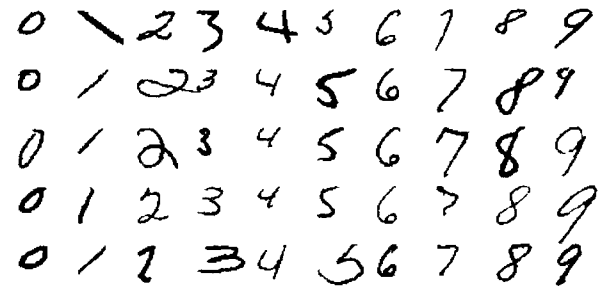

Figure 4: Sample data for experiments.

In common with all other search algorithms, a GA performs a search of a multidimensional space containing a hypersurface known as the fitness surface. A particular parameter set defines a point on a hyperplane on to which the surface is projected, with the height of the surface above the hyperplane reflecting the relative merit of the problem solution represented by the parameter set.

The basis of a GA is that a population of problem solutions is maintained in the form of chromosomes, which are strings encoding problem solutions. Strings can be binary or *have many possible alternatives (genes) at each position*. The strings are converted into problem solutions, which are then evaluated according to an objective scoring function. Often it is not possible to exhaustively test all aspects of a solution, and noise may be present on the objective function, so the assigned fitness is an estimate of the true fitness of a chromosome. It is important that this is a good estimate, otherwise the selective pressure that favors truly high scoring chromosomes can be lost in the noise caused by poor fitness estimates.

Following fitness evaluation, a new population of chromosomes is generated by applying a set of genetic operators to the original population. These are basically random copying and altering of individuals from the original population with the probability of copying of any individual from one generation to the next being proportional to its fitness. During the copying process two operations may be performed—a gene may be erroneously copied (mutation), or a new individual may be formed by combining segments from two chromosomes by copying one chromosome up to a specific location on the chromosome, then copying a different chromosome (crossover).

To give an idea of how the framework yields better system, we have utilized the Choquet integral to combine the outputs of the networks with importance of each network, which is assigned by genetic algorithm. Figure 3 shows the GA procedure for the proposed method. The GA approach to our problem takes pieces of the fuzzy density values to combine neural networks as such strings as shown in Figure 3. In the following, $\hat{g}_\lambda(A)$ and $\hat{g}_i$ denote the human-provided values, and $g_\lambda(A)$ and g_i denote the identified values.

In this method, chromosomes encode the fuzzy density values g_i^j by a vector $C_j = (g_1^j, g_2^j, \ldots, g_k^j; \lambda_j)$. The fitness function $f(C_j)$ for chromosome C_j is the sum of the differences between human-provided fuzzy measure value $\hat{g}_\lambda(A)$ and fuzzy measure value obtained by g_i^j and λ_j.

$$f(C_j) = \sum_{A \in B(X)} |\hat{g}_\lambda(A) - \frac{1}{\lambda_j}\left[\prod_{x_i \in A}(1 + \lambda_j g_i^j) - 1\right]| \tag{9}$$

With these the genetic operators yield an optimal set of parameters to combine neural networks. When adopting GA to solve a problem at hand, we have to consider about the computational cost and convergence problem: It may not converge if the problem is structured poorly. However, in the hybrid system, they are not so serious because the solution space is not that large, and moreover, the GA replaces a moderate amount of trial-and-error overhead we have to put in the course of deciding the optimal parameters.

5 Experiments and Analysis

5.1 Environments

In the experiments, we have used the handwritten digit database of Concordia University of Canada, which consists of 6000 unconstrained numerals originally collected from dead letter envelopes by the U.S. Postal Services at different locations in the U.S. The numerals of this database were digitized in bilevel on a 64×224 grid of 0.153mm square elements, giving a resolution of approximately 166 PPI. Among the data, 4000 numerals were used for training and 2000 numerals for testing. Figure 4 shows some representative samples taken from the database. We can see that many different writing styles are apparent, as well as numerals of different sizes and stroke widths.

Digits, whether handwritten or typed, are essentially line drawings, i.e., one-dimensional structures in a two-dimensional space. Thus, local detection of line segments seems to be an adequate feature extraction method. For each location in the image, information about the presence of a line segment of a given direction is stored in a feature map [3]. Especially, in this paper Kirsch masks have been used for extracting directional features.

Since the data set was prepared by thorough preprocessing, in this paper, each digit is scaled to fit in a 16×16 bounding box such that the aspect ratio of the image is preserved. Then, feature vectors for horizontal, vertical, right-diagonal, and left-diagonal directions are obtained from the scaled image. The final step in extracting the features compresses each 16×16 directional vector into 4×4 vector with averaging operator, which produces a value for 2×2 pixels with dividing by 4 the value obtained by summing the four values. Moreover, 4×4 compressed image can be considered as a good candidate for global features. In addition to those two features, we have also used a contour feature: 15 complex Fourier descriptors from the outer contours and simple topological features from the inner contours. As a result, available features include four 4×4 local features, one 4×4 global features, and structural features extracted from the contours of the numerals.

5.2 Experimental Results

To evaluate the performance of the combining methods, we have implemented three different networks, each of which is a two-layer neural network using different features. NN_1, NN_2 and NN_3 are the networks using normalized image, Kirsch features, and sequence of contour features, respectively. In this way each network makes the decision through its own criterion. Each of the three networks was trained with 4000 samples, and tested on 2000 samples from the Concordia database.

The error backpropagation algorithm was used for the training and the iterative estimation process was stopped when an average squared error of 0.9 over the training set was obtained, or when the number of iteration reaches 1000, which was adopted mainly for preventing networks from overtraining. The parameter values used for training were: learning rate is 0.4 and momentum parameter is 0.6. An input vector is classified as belonging to the output class associated with the highest output activation.

For the use of weighting among the networks, we assigned the degree of importance of each network, g^i, based on how good these networks performed on the train data. We computed these values as follows:

$$g^i = \frac{p_i}{\sum_j p_j} \cdot d\text{sum}, \qquad (10)$$

Table 1: Results of combining networks by the fuzzy integral on three different networks.

	Actual	Partial decision			Fuzzy
	class	NN_1	NN_2	NN_3	decision
1	6	6 (0.99)	6 (0.99)	5 (0.33)	6 (0.68)
2	8	8 (0.99)	0 (0.00)	8 (0.99)	8 (0.66)
3	2	2 (0.99)	2 (0.99)	2 (0.99)	2 (0.99)
4	7	8 (0.01)	8 (0.00)	7 (0.96)	7 (0.32)
5	9	9 (0.99)	9 (0.99)	8 (0.07)	9 (0.66)
6	3	3 (0.99)	3 (0.99)	3 (0.99)	3 (0.99)
7	7	8 (0.03)	0 (0.04)	7 (0.48)	7 (0.16)
8	5	5 (0.93)	3 (0.13)	3 (0.62)	5 (0.32)
9	2	8 (0.34)	2 (0.99)	8 (0.20)	2 (0.33)
10	9	9 (0.99)	1 (0.99)	9 (0.99)	9 (0.67)
11	4	4 (0.99)	4 (0.99)	4 (0.99)	4 (0.99)
12	8	8 (0.99)	0 (0.98)	8 (0.83)	8 (0.62)

Table 2: The result of recognition rates (%).

Methods	Correct	Error	Reject
NN_1	89.05	7.00	3.95
NN_2	95.40	3.75	0.85
NN_3	93.95	4.10	1.95
NN+Fuzzy	97.78	1.90	0.32
NN+Fuzzy+GA	98.05	1.95	0.00

where p_i is the performance of network NN_i for the train data and $d$sum is the desired sum.

Table 1 reports the results of combining networks by the fuzzy integral on three different networks. In this table the value in the parentheses represent the confidence of the evaluation result. As can be seen, cases 1 and 2 are misclassified by NN_3 and NN_2, respectively. However, in the final evaluations they are correctly classified. In cases 4 and 9, one network with strong evidence overwhelmed the other networks, producing correct classification. Furthermore, in case 7, the fuzzy integral made a correct decision despite that the partial decisions from the individual neural networks are totally inconsistent. The effect of misclassification by the other networks has given rise to small fuzzy integral values for the correct classification in this case.

Figure 5 shows the best and average fitness changes in the genetic algorithm based method. As the figure depicts, it is clear that the performance increases as the generation goes and after the initial radical improvements the overall fitness settles down soon. Table 2 reports the recognition rates with respect to the three different networks and their combinations by utilizing combining methods. NN_{all} here means the network trained with all the available features.

As can be seen, any method of combining neural networks produces better results than individual networks, and the overall classification rates for the soft-

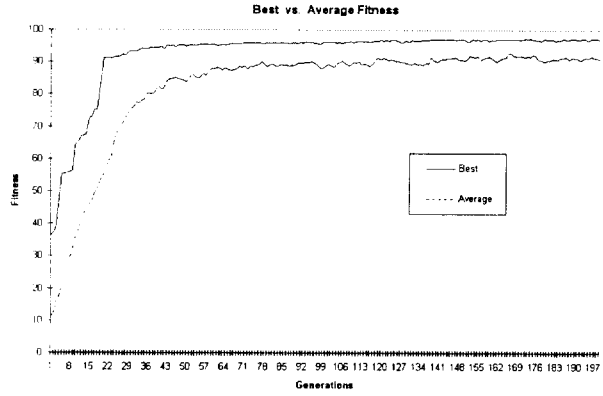

Figure 5: Best and average fitness changes as generation goes.

Table 3: Comparisons of the presented method with the related (%).

Methods	Correct	Error	Reject
Lam et al. (1988)	93.10	2.95	3.95
Nadal et al. (1988)	86.05	2.25	11.70
Legault et al. (1989)	93.90	1.60	4.50
Krzyzak et al. (1990)	86.40	1.00	12.60
Krzyzak et al. (1990)	94.85	5.15	0.00
Mai et al. (1990)	92.95	2.15	4.90
Suen et al. (1990)	93.05	0.00	6.95
Kim et al. (1994)	95.85	4.15	0.00
The proposed method	98.05	1.95	0.00

computing techniques are higher than those for other consensus methods. Although the network learned the training set almost perfectly in all three cases, the performances on the test sets are quite different. Furthermore, we can see that the performance did not improve by training a large network with considering all the features used by each network. This is a strong evidence that multiple neural network might produce better result than conventional single network approach. Actually, the proposed softcomputing methods have a small, but statistically significant ($p > 0.995$), advantage in recognition rates obtained by the conventional methods.

To give a fairer view of the performance in this field of handwritten digit recognition, Table 3 shows the performances of the presented hybrid method along with the results reported by some previous methods in the literature. It also provides information about the size of the data sets used for training and testing along with the scanning resolution in PPI. The error rate of the proposed method is 1.95%, which is a big improvement compared with those of the previous methods. Actually, this performance is remarkable and can stand comparison with the best results reported in the literature.

6 Concluding Remarks

This paper has presented a softcomputing framework for producing an improved performance on real-world classification problem, especially handwritten numeral recognition. One of the important advantages of the method based on softcomputing techniques is that not only is the classification results combined but that the relative importance of the different networks is also considered. The experimental results for classifying a large set of handwritten numerals show that it improves the generalization capability significantly. This indicates that even these straightforward, computationally tractable approach can significantly enhance pattern recognizer. The complementary nature of neural networks, fuzzy logic and genetic algorithms leads us to believe that a further refined genetic neuro-fuzzy system will significantly improve the state-of-the-art pattern recognizers.

References

[1] T. Fukuda, "Fuzzy-neuro-GA based intelligent robotics," *Computational Intelligence: Imitating Life*, J.M. Zurada, R.J. Marks II, C.J. Robinson, Ed., IEEE Press, 252–263, 1994.

[2] L.K. Hansen and P. Salamon, "Neural network ensembles," *IEEE Trans. Patt. Anal. Mach. Inte.*, **12**, 993–1001, 1990.

[3] S. Knerr, L. Personnaz and G. Dreyfus, "Handwritten digit recognition by neural networks with single-layer training," *IEEE Trans. on Neural Networks*, **3**(6), 962–968, 1992.

[4] T. Murofushi and M. Sugeno, "An interpretation of fuzzy measure and the Choquet integral as an integral with respect to a fuzzy measure," *Fuzzy Sets and Systems*, **29**, 201–227, 1989.

[5] M.P. Perrone and L.N. Cooper, "When networks disagree: Ensemble methods for hybrid neural networks," *Neural Net. for Speech and Image Proc.* Mammone, R.J. ed., Chapman-Hill, London, 1993.

[6] M. Srinivas and L.M. Patnaik, "Genetic algorithms: a survey," *IEEE Computer*, 17–26, June 1994.

[7] H. Tahani and J.M. Keller, "Information fusion in computer vision using the fuzzy integral," *IEEE Trans. Syst. Man. Cyber*, **20**, 733–741, 1990.

[8] H. Takagi, "Fusion technology of fuzzy theory and neural networks," In *Proc. Int. Conf. Fuzzy Logic & Neural Networks*, 13–26, 1990.