NORTH-HOLLAND

Fuzzy Aggregation of Modular Neural Networks With Ordered Weighted Averaging Operators

Sung-Bae Cho
Department of Computer Science, Yonsei University, Seoul, South Korea

ABSTRACT

This paper presents an efficient fuzzy neural system which consists of modular neural networks combined by the fuzzy integral with ordered weighted averaging (OWA) operators. The ability of the fuzzy integral to combine the results of multiple sources of information has been established in several previous works. The key point of this paper is to formalize modular neural networks as information sources, and show the feasibility of the fuzzy integral extended by OWA operators in the problem of combining neural outputs, especially in the case that the networks differ substantially from each other in accuracy. The experimental results with the recognition problem for on-line handwritten characters show that the performance of individual networks is improved significantly.

KEYWORDS: ***fuzzy neural system, modular neural networks, fuzzy integral, OWA operators, character recognition***

1. INTRODUCTION

In the past several years, there has been tremendous growth in the complexity of the recognition, estimation, and control problems expected to be solved by neural networks. In solving these problems, 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-takes-all cross-validatory model selection. However, recent

Address correspondence to Professor Sung-Bae Cho, Department of Computer Science, Yonsei University, 134 Shinchon-dong, Sudaemoon-ku, Seoul 120-749, South Korea.

Received October 1994; accepted May 1995.

International Journal of Approximate Reasoning 1995; 13:359–375

655 Avenue of the Americas, New York, NY 10010
0888-613X/95/$9.50
SSDI 0888-613X(95)00059-P

theoretical and experimental work indicates that we can improve performance by considering methods for combining neural networks [1–6].

There have been proposed various neural-network optimization methods based on combining estimates, such as boosting, competing experts, ensemble averaging, metropolis algorithms, stacked generalization, and stacked regression. A general result from the previous works is that averaging separate networks improves generalization performance for the mean squared error. If we have networks of different accuracy, however, it is obviously not good to take their simple average or use simple voting.

To give a solution to the problem, we developed a fusion method that considers the difference of performance of each network on combining the networks, and that is based on the notion of fuzzy logic, especially the fuzzy integral [7, 8]. This method combines the outputs of separate networks with the importance of each network, which is subjectively assigned as usual in fuzzy logic. In this paper, we extend the structure of the fuzzy integral with ordered weighted averaging (OWA) operators [9] and apply the method to integrating modular neural networks.

OWA operators have the property of lying between the AND, requiring all the criteria to be satisfied, and the OR, requiring at least one of the criteria to be satisfied. They are different from the classical weighted average in that coefficients are not associated directly with a particular attribute, but rather with an ordered position [10]. Furthermore, the structure of these operators is very much in the spirit of combining the criteria under the guidance of a quantifier. The last part of this paper will demonstrate the effectiveness of the method by experimental results on a difficult optical-character-recognition problem.

The rest of this paper is organized as follows. Section 2 formulates the problem of combining modular neural networks, and shows how it might generate better results. In Section 3, we introduce the fuzzy integral for combining the modular neural networks, and extend it with OWA operators. Shown in Section 4 is a simple example to give an account of how the proposed method works out. Finally, Section 5 shows the results with the recognition of on-line handwritten characters.

2. FORMULATION OF MODULAR NEURAL NETWORKS

In this section, we present the modular neural network (MNN) which combines a population of neural network outputs to estimate a function

[1] The outputs of neural networks are not just likelihoods or binary logical values near zero or one. Instead, they are estimates of Bayesian *a posteriori* probabilities of a classifier.

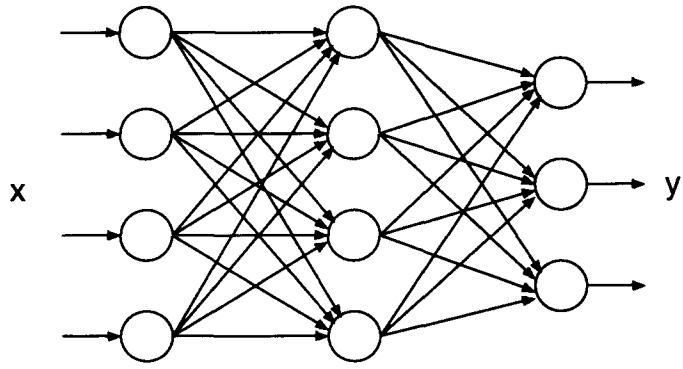


Figure 1. A two-layered neural-network architecture.

$f(x)$ defined by $f(x) = E[y \mid x]$ [11].[1] Figure 1 shows a two-layered neural network. 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 $X = (x_1, x_2, \cdots, x_T)$ and the output set $\Omega = \{\omega_1, \omega_2, \cdots, \omega_c\}$, each output neuron estimates the probability $P(\omega_i \mid X)$ of belonging to this class by

$$P(\omega_i \mid X) \approx f\left\{ \sum_k w_{ik}^{om} f\left(\sum_j w_{kj}^{mi} x_j \right) \right\},$$

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 of P is selected as the corresponding class.

The basic idea of the modular neural network here is to develop n independently trained neural networks with relevant features, and to classify a given input pattern by obtaining a combination from each copy of the network and then deciding the collective classification by utilizing combination methods [1, 12] (see Figure 2). In the following, we shall sketch how the modular neural network scheme generates an improved regression estimate [6].

Suppose that we have two finite data sets whose elements are all independent and identically distributed random variables: a training data set $A = \{(x_m, y_m)\}$ and a cross-validatory data set $CV = \{(x_l, y_l)\}$. Further suppose that we have used A to generate a set of functions, $F = f_i(x)$,

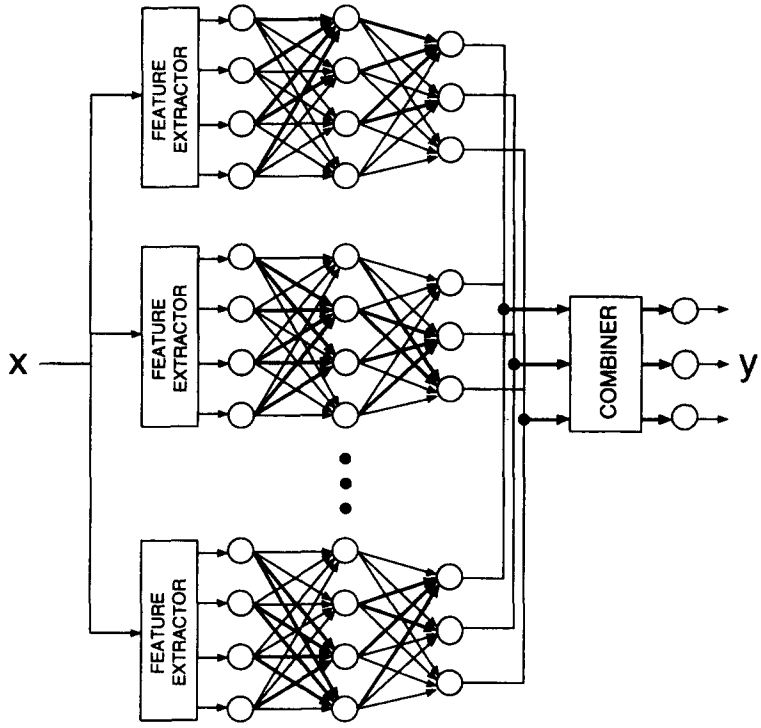


Figure 2. The modular neural-network scheme.

each element of which approximates $f(x)$. We would like to show that the MNN estimator, $f_{\text{MNN}}(x)$, produces an improved approximation to $f(x)$.

Define the misfit of the function $f_i(x)$, its deviation from the true solution, as $m_i(x) \equiv f(x) - f_i(x)$. The mean squared error can now be written in terms of $m_i(x)$ as

$$\text{MSE}[f_i] = E\left[m_i^2\right].$$

The average mean squared error is therefore

$$\overline{\text{MSE}} = \frac{1}{n} \sum_{i=1}^{n} E\left[m_i^2\right].$$

Define the MNN regression function $f_{\text{MNN}}(x)$ as

$$f_{\text{MNN}}(x) \equiv \frac{1}{n} \sum_{i=1}^{n} f_i(x) = f(x) - \frac{1}{n} \sum_{i=1}^{n} m_i(x).$$

If we now assume that the $m_i(x)$ are mutually independent with zero mean, we can calculate the mean squared error of $f_{\text{MNN}}(x)$ as

$$\begin{aligned}
\text{MSE}[f_{\text{MNN}}] &= E\left[\left(\frac{1}{n}\sum_{i=1}^{n} m_i\right)^2\right] \\
&= \frac{1}{n^2}E\left[\sum_{i=1}^{n} m_i^2\right] + \frac{1}{n^2}E\left[\sum_{i\neq j} m_i m_j\right] \\
&= \frac{1}{n^2}E\left[\sum_{i=1}^{n} m_i^2\right] + \frac{1}{n^2}\sum_{i\neq j} E[m_i]E[m_j] \\
&= \frac{1}{n^2}E\left[\sum_{i=1}^{n} m_i^2\right],
\end{aligned}$$

which implies that

$$\text{MSE}[f_{\text{MNN}}] = \frac{1}{n}\overline{\text{MSE}}.$$

This is a powerful result because it tells us that by averaging regression estimates, we can reduce our mean squared error by a factor of n with respect to the population performance.

3. FUZZY AGGREGATION OF NEURAL NETWORKS

The fuzzy integral introduced by Sugeno and the associated fuzzy measures provide a useful way for aggregating information [13]. The ability of the fuzzy integral to combine the results of multiple sources of information has been established in several previous works [9, 14, 15]. In the following we shall introduce some definitions of it and present an effective method for combining the outputs of multiple networks with regard to subjectively defined importances of individual networks.

DEFINITION 1 *A set function* $g: 2^X \to [0, 1]$ *is called a fuzzy measure if*

(1) $g(\varnothing) = 0, g(X) = 1$,

(2) $g(A) \leq g(B)$ *if* $A \subset B$,

(3) *If* $\{A_i\}_{i=1}^{\infty}$ *is an increasing sequence of measurable sets, then*

$$\lim_{i\to\infty} g(A_i) = g\left(\lim_{i\to\infty} A_i\right).$$

DEFINITION 2 *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*

$$h(x) \circ g(\cdot) = \max_{E \subseteq X} \left[\min \left(\min_{x \in E} h(x), g(E) \right) \right].$$

The following properties of the fuzzy integral can be easily proved [15].

1. *If $h(x) = c$ for all $x \in X$, $0 \leq c \leq 1$, then*

$$h(x) \circ g(\cdot) = c.$$

2. *If $h_1(x) \leq h_2(x)$ for all $x \in X$, then*

$$h_1(x) \circ g(\cdot) \leq h_2(x) \circ g(\cdot).$$

3. *If $\{A_i \mid i = 1, \cdots, n\}$ is a partition of the set X, then*

$$h(x) \circ g(\cdot) \geq \max_{i=1}^{n} e_i,$$

where e_i is the fuzzy integral of h with respect to g over A_i.

The calculation of the fuzzy integral with respect to a g_λ-fuzzy measure would only require the knowledge of the density function, where the ith density, g^i, is interpreted as the degree of importance of the source y_i towards the final evaluation. These densities can be subjectively assigned by an expert, or can be generated from data. The value obtained from comparing the evidence and the importance using the min operator is interpreted as the grade of agreement between the real possibilities $h(y)$ and the expectations g. Hence fuzzy integration is interpreted as searching for the maximal grade of agreement between the objective evidence and the expectation. For further information on the fuzzy integral for network fusion, refer to [7, 8].

Let $\Omega = \{\omega_1, \omega_2, \cdots, \omega_c\}$ be a set of classes of interest. Note that each ω_i may, in fact, be a set of classes itself. Let $Y = \{y_1, y_2, \cdots, y_n\}$ 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 classifying the object A in class ω_k using the network y_i, where 1 indicates absolute certainty that the object A is really in class ω_k, and 0 implies absolute certainty that A is not in ω_k.

In [9] Yager extended the fuzzy integral with two special families of OWA operators, S-OWA-AND and S-OWA-OR.

DEFINITION 3 *A mapping F from*

$$I^n \to I \qquad (\textit{where} \quad I = [0,1])$$

is called an OWA operator of dimension n if associated with F is a weighting vector

$$W = \begin{bmatrix} W_1 \\ W_2 \\ \vdots \\ W_n \end{bmatrix}$$

such that
(1) $W_i \in (0, 1)$,
(2) $\sum_i W_i = 1$,
and where

$$F(a_1, a_2, \cdots, a_n) = W_1 b_1 + W_2 b_2 + \cdots + W_n b_n,$$

where b_i *is the ith largest element in the collection* $a_1, a_2, \cdots, a_n$.

In [10] Yager shows how different assignments of the weights allow implementation of different quantifiers. For example, W^*, with $W_1 = 1$ and $W_i = 0$, $i \neq 1$, provides the max operator. W_* with $W_n = 1$ and $W_i = 0$ or $i \neq n$ gives us the min operator. Finally, $W_i = 1/n$ gives us the average $(1/n) \sum a_i$. This shows that the more weights are near the bottom, the more AND-like the aggregation, and the more the weights are near the top, the more OR-like the aggregation.

There are two special families of OWA operators which are useful for extending the fuzzy integral [9]. These are called the S-OWA-AND and S-OWA-OR operators. The S-OWA-AND operators are defined by

$$\hat{F}_\alpha(a_1, \cdots, a_n) = \frac{1 - \alpha}{n} \sum_i a_i + \alpha \min_i a_i.$$

They provide for AND-like aggregations. In the formulation for the fuzzy integral we can obtain the effect of S-OWA-AND operators by replacing $\min_{x \in E} h(x)$ with

$$\frac{1 - \alpha}{\text{Card } E} \sum_{x \in E} h(x) + \alpha \min_{x \in E} h(x).$$

The parameter α lies in the unit interval. The closer α is to one, the more AND-like the aggregation becomes.

On the contrary, the S-OWA-OR operator provides for an OR-like aggregation. This operator is defined by

$$\tilde{F}_\beta(a_1,\cdots,a_n) = \frac{1-\beta}{n}\sum_i a_i + \beta \max_i a_i.$$

Here again the parameter β lies in the unit interval and the closer β is to 1, the more like a pure OR the operation. This S-OWA-OR operator can be used to provide a further generalization of the fuzzy integral. Let us denote $\min(\min_{x\in E} h(x), g(E))$ as $H(E)$. The advantage of the fuzzy integral is in requiring that *at least one* subset E of X satisfy $H(E)$. With n the cardinality of X we can change the aggregation to

$$\frac{1-\beta}{2^n}\sum_{E\subset X} H(E) + \beta \max_{E\subset X} H(E).$$

With this change, depending on the choice of β, we are requiring that *some* or *a few* of the E satisfy $H(E)$ rather than just one.

4. A SIMPLE EXAMPLE

To get an idea of how the consensus decision is produced by the fuzzy integral, let us consider a simple example of discriminating two class patterns. Suppose that we have implemented three different neural networks of $g^1 = 0.34$, $g^2 = 0.32$, and $g^3 = 0.33$, and obtained the network outputs for an input image as follows:

$$\mathrm{NN}_1 : (0.6, 0.8),$$
$$\mathrm{NN}_2 : (0.7, 0.3),$$
$$\mathrm{NN}_3 : (0.1, 0.4).$$

In the case of using a voting method, the final decision is for class 2, because NN_1 and NN_3 vote for class 2 while only NN_2 votes for class 1. The majority voting rule chooses the classification made by more than half the networks. When there is no agreement among more than half the networks, the result is considered an error.

Another simple approach to combine the results on the same X from all three networks is to use the average value as a new estimation of combined network:

$$P(\omega_i \mid X) = \frac{1}{3}\sum_{k=1}^{3} P_k(\omega_i \mid X), \qquad \text{where} \quad i = 1, 2.$$

In this example, the average method produces class 2 as the correct class, since the output for class 2 is 0.5 [(0.8 + 0.3 + 0.4)/3] while that for class 1 is 0.47 [(0.6 + 0.7 + 0.1)/3]. On utilizing the weighted-average method, class 2 is also chosen as the correct class, because class 2 yields 0.5 (0.34 × 0.8 + 0.32 × 0.3 + 0.33 × 0.4) whereas class 1 yields 0.46 (0.34 × 0.6 + 0.32 × 0.7 + 0.33 × 0.1).

Now consider the case of using the fuzzy integral. First of all, we have to calculate the λ value from the g values. The Sugeno measure g must have a parameter λ satisfying $0.0359\lambda^2 + 0.3266\lambda - 0.001 = 0$ (refer to [8]). The unique root greater than -1 for this equation is $\lambda = 0.0305$, which produces the following fuzzy measure on the power set of $Y = \{y_1, y_2, y_3\}$:

Subset A	$g_{0.0305}(A)$
$\varnothing$	0
$\{y_1\}$	0.34
$\{y_2\}$	0.32
$\{y_3\}$	0.33
$\{y_1, y_2\}$	0.6633
$\{y_2, y_3\}$	0.6532
$\{y_1, y_3\}$	0.6734
$\{y_1, y_2, y_3\}$	1.0

As expected, the subset of criteria $\{y_1, y_3\}$ is more important for confirming the hypothesis than either subset $\{y_1, y_2\}$ or $\{y_2, y_3\}$. The following table shows how the consensus is formed, where $H(E) = \min(h(y_i), g(A_i))$. In this table $h(y_1) = 0.6$, $h(y_2) = 0.7$, and $h(y_3) = 0.1$ for class 1, and $h(y_1) = 0.8$, $h(y_2) = 0.3$, and $h(y_3) = 0.4$ for class 2:

Class	$h(y_i)$	$g(A_i)$	$H(E)$	max[$H(E)$]
1	0.7	$g(\{y_2\}) = g^2 = 0.32$	0.32	
	0.6	$g(\{y_2, y_1\}) = g^2 + g^1 + \lambda g^2 g^1 = 0.66$	0.6	✓
	0.1	$g(\{y_2, y_1, y_3\}) = 1.0$	0.1	
2	0.8	$g(\{y_1\}) = g^1 = 0.34$	0.34	
	0.4	$g(\{y_1, y_3\}) = g^1 + g^3 + \lambda g^1 g^3 = 0.67$	0.4	✓
	0.3	$g(\{y_1, y_3, y_2\}) = 1.0$	0.3	

Finally, class 1 is selected as output. This example shows how the small differences in g^i conspire to dramatically change the performance from

that of simple averaging. The following table shows how the classification result depends on the values of the g^i's:

Case	g^1	g^2	g^3	Classification
1	0.1	0.2	0.3	Class 2
2	0.1	0.3	0.2	Class 1
3	0.2	0.1	0.3	Class 2
4	0.2	0.3	0.1	Class 1
5	0.3	0.1	0.2	Class 1
6	0.3	0.2	0.1	Class 1

Furthermore, if we extend the fuzzy integral with the OWA operators, we can get somewhat different results from the usual fuzzy integral. If we adopt the S-OWA-AND operator, $\min_{x \in E} h(x)$ is replaced with the following $\hat{h}(y_i)$'s:

$$\hat{h}(y_0) = (1 - \alpha) \times h(y_0) + \alpha \times h(y_0),$$

$$\hat{h}(y_1) = (1 - \alpha) \times \frac{h(y_0) + h(y_1)}{2} + \alpha \times h(y_1),$$

$$\hat{h}(y_2) = (1 - \alpha) \times \frac{h(y_0) + h(y_1) + h(y_2)}{3} + \alpha \times h(y_2).$$

Assume that $\alpha = 0.1$. Then we can obtain the following result:

Class	$\hat{h}(y_i)$	$g(A_i)$	$H(E)$	max[$H(E)$]
1	$0.9 \times 0.7 + 0.1 \times 0.7 = 0.7$	0.32	0.32	
	$0.9 \times (0.7 + 0.6)/2 + 0.1 \times 0.6 = 0.645$	0.66	0.645	✓
	$0.9 \times (0.7 + 0.6 + 0.1)/3 + 0.1 \times 0.1 = 0.43$	1.0	0.43	
2	$0.9 \times 0.8 + 0.1 \times 0.8 = 0.8$	0.34	0.34	
	$0.9 \times (0.8 + 0.4)/2 + 0.1 \times 0.4 = 0.58$	0.67	0.58	✓
	$0.9 \times (0.8 + 0.4 + 0.3)/3 + 0.1 \times 0.3 = 0.48$	1.0	0.48	

Class 1 is also selected as the final output in this case. Further extending it with the S-OWA-OR operator, $H(E)$ becomes the following:

$$\tilde{H}(E) = (1 - \beta) \times \frac{H(E_0) + H(E_1) + H(E_2)}{8} + \beta \max H(E).$$

If $\beta = 0$, then for class 1,

$$\tilde{H}_1(E) = 1 \times \frac{0.32 + 0.645 + 0.43}{8} + 0 \times 0.645 = 0.174,$$

while for class 2,

$$\tilde{H}_2(E) = 1 \times \frac{0.34 + 0.58 + 0.48}{8} + 0 \times 0.58 = 0.175.$$

Thus, class 2 is chosen as the output. Notice that the classification result becomes different from the usual fuzzy integral. However, if $\beta = 0.1$, for class 1

$$\tilde{H}_1(E) = 0.9 \times \frac{0.32 + 0.645 + 0.43}{8} + 0.1 \times 0.645 = 0.221,$$

whereas for the class 2

$$\tilde{H}_2(E) = 0.9 \times \frac{0.34 + 0.58 + 0.48}{8} + 0.1 \times 0.58 = 0.215.$$

In this case, class 1 is selected as the output.

This section has given accounts of the characteristics of the proposed method by using a simple example.

5. EXPERIMENTAL RESULTS

In the experiment, handwritten characters were input to the computer (a Sun workstation) by an LCD tablet of Photron FIOS-6440 which samples 80 dots per second. The tasks were to classify Arabic numerals, uppercase letters, and lowercase letters which were collected from 13 writers. An input character consists of a set of strokes, each of which begins with a pen-down movement and ends with a pen-up movement. Several preprocessing algorithms were applied to successive data points in a stroke to reduce quantization noise and fluctuations of the writer's pen motion. Data points, representing a single character, were resampled with a fixed number of regularly spaced points. Then, a sequence of preprocessed data points was approximated by a sequence of 8-directional straight-line segments—the chain code, as used by Freeman [16].

To evaluate the performance of the presented method, we implemented three different networks, each of which is a two-layered neural network having a different number of input neurons and 20 hidden neurons. NN_1, NN_2, and NN_3 have 10, 15, and 20 input neurons, respectively. In each

case, the network makes a decision based on its resolution. For example, NN_1 uses sparsely sampled inputs, and in doing so is able to overcome variations in input noise. NN_3, by comparison, uses a finer view of the input image.

Each of the three networks was trained by the error back-propagation algorithm with 40 samples per class, validated with another 500 samples, and tested on 10 sets of additional samples collected from 10 different writers. The training process was stopped when the recognition rate over the validation set was optimized. This process and early stopping mechanism were adopted mainly to prevent networks from overtraining. The initial parameter values used for training were: learning rate 0.4 and momentum parameter 0.6. An input vector is classified as belonging to the output class associated with the highest output activation. Each of the following experiments consisted of 10 trials in which the different data were made from different writers.

We assigned the fuzzy densities g^i, the degree of importance of each network, based on how well these networks performed on validation data. We computed these values as follows:

$$g^i = \frac{p_i}{\sum_j p_j} d_{\text{sum}},$$

where p_i is the performance of network NN_i for the validation data and d_{sum} is the desired sum of fuzzy densities.

Table 1 shows the confusion matrices of the network outputs for the numeral recognition task, where the combined result is made by the fuzzy integral. The performance for the combined outputs is much better than for either of the individual networks, leading to a significant reduction in error rate. We also see a strongly diagonal matrix for the combined output, indicating the complementary nature of the confusions made by the individual networks.

Table 2 shows the recognition rates of numerals, uppercase letters, and lowercase letters with respect to the three different networks and their combinations by utilizing consensus methods such as majority voting, averaging, and the fuzzy integral. All results are averaged over 10 different sets of the data. In this table, NN_1 to NN_3 represent the three individual networks, and NN_{all} a large network trained with all the features used by each network.

Although the network learned the training set almost perfectly in all three cases, the performance on the test sets is quite different. Furthermore, we can see that the performance did not improve significantly on training a large network that considered all the features used by each network. This is a strong evidence that a modular neural network might

Table 1. Confusion Matrices of the Three Individual Networks and the Combined Output for the Numerical Recognition Task[a]

(a) Network 1

	0	1	2	3	4	5	6	7	8	9
0	30	0	0	0	1	0	12	2	5	0
1	0	27	0	2	1	0	4	2	0	1
2	1	2	38	0	1	0	0	0	1	0
3	2	3	0	39	0	3	0	1	3	0
4	0	0	1	0	48	0	1	1	1	2
5	1	0	0	3	0	42	0	0	2	0
6	1	0	0	0	0	0	41	0	0	0
7	1	4	0	7	2	0	0	36	1	0
8	1	0	0	0	3	2	0	1	61	0
9	0	2	0	2	0	1	0	4	0	47

(b) Network 2

	0	1	2	3	4	5	6	7	8	9
0	24	0	0	0	5	2	6	0	12	1
1	2	25	0	2	5	0	0	3	0	0
2	0	0	41	0	1	0	0	0	1	0
3	0	0	0	46	0	2	0	0	2	1
4	0	0	1	1	45	0	1	0	0	6
5	1	0	0	0	2	37	3	0	4	1
6	0	0	1	0	0	0	39	0	1	1
7	0	3	1	5	4	0	0	37	0	1
8	5	0	0	1	0	1	2	1	58	0
9	0	1	0	2	0	5	0	1	1	46

(c) Network 3

	0	1	2	3	4	5	6	7	8	9
0	24	5	1	0	1	2	11	1	5	0
1	0	27	0	1	4	0	1	2	2	0
2	0	2	39	0	0	0	1	0	1	0
3	1	2	0	41	1	0	1	0	5	0
4	1	0	0	0	42	1	1	2	4	3
5	1	0	0	2	0	44	0	0	1	0
6	3	1	1	0	2	0	34	0	0	1
7	0	7	1	6	0	1	1	34	1	0
8	5	0	1	0	0	1	1	1	59	0
9	3	1	0	1	0	1	0	2	7	41

Table 1. *continued.*

(d) Combined output

	0	1	2	3	4	5	6	7	8	9
0	32	0	0	0	2	0	10	1	5	0
1	0	28	0	2	1	0	3	2	0	1
2	0	1	42	0	0	0	0	0	0	0
3	2	4	0	43	0	2	0	0	0	0
4	0	0	1	0	47	0	1	0	2	3
5	1	0	0	1	0	44	0	0	2	0
6	1	0	0	0	0	0	41	0	0	0
7	1	4	0	6	2	0	0	38	0	0
8	4	1	0	0	0	0	0	0	63	0
9	0	3	0	3	0	2	0	2	0	46

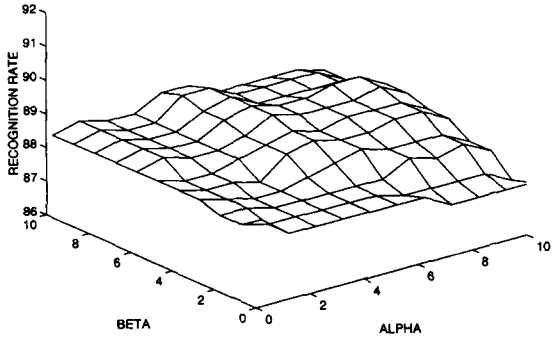
[a] Each vertical column is labeled by the target output, and each horizontal row represents an output by the network.

produce better results than a conventional single network. The fuzzy-integral approach has a statistically significant ($p > 0.999$) advantage in recognition rate over the individual neural networks and other aggregation methods. The statistical comparison is based on a paired-sample t test with 10 degrees of freedom. It is also seen that averaging is better than voting for the problem at hand.

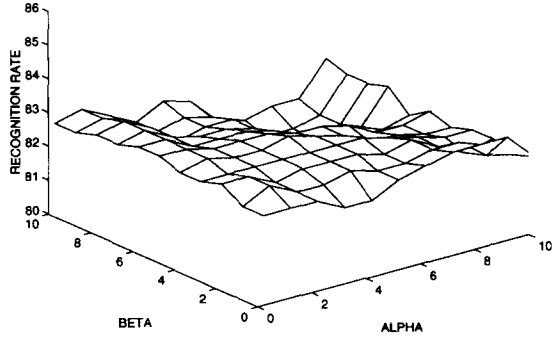
Figure 3 shows the recognition rates of the presented method with the OWA operators for the three tasks. The results indicate that the performance of the fuzzy integral might be enhanced if we selected the parameters α and β appropriately. How to determine the parameters depends

Table 2. Means and Standard Deviations of Recognition Rates (%)

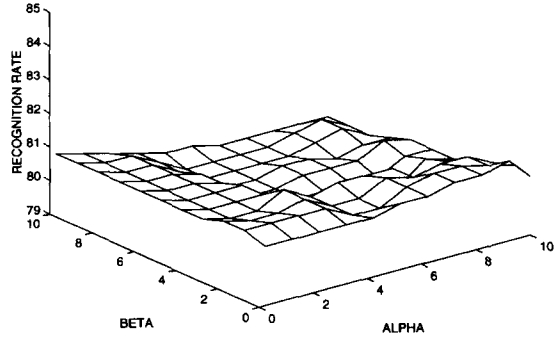
Nets	Numeral		Uppercase		Lowercase	
	Mean	S.D.	Mean	S.D.	Mean	S.D.
NN_1	82.6	6.36	73.2	8.95	73.9	7.73
NN_2	81.2	7.16	68.6	9.14	71.8	8.86
NN_3	81.0	7.15	70.8	10.60	72.1	9.30
NN_{all}	77.6	6.31	72.1	8.93	74.7	10.01
Voting	84.9	8.31	74.0	9.28	74.6	7.97
Average	86.9	7.24	75.2	9.95	78.2	8.85
Fuzzy	88.1	7.14	76.1	9.85	80.3	7.24
Fuzzy with OWA	88.9	7.25	76.7	9.92	81.9	7.83



(a) Numerals



(b) Uppercases



(c) Lowercases

Figure 3. The recognition rates with OWA operators: (a) numerals, (b) uppercase, (c) lowercase.

largely on the problem at hand, but we can infer the following rules of thumb from the experiments:

- If the recognition rates of individual networks are high, then choose large values for the parameters.
- It is not good to choose α and β large simultaneously.

6. CONCLUSIONS

Modular neural networks aggregated by the fuzzy integral represent a powerful recognizer which produces improved performance on real-world classification problems, in particular handwritten-character recognition. This indicates that even this straightforward, computationally tractable approach can significantly enhance pattern recognition. Future efforts will concentrate on refining the feature extraction to capture more information, and testing the efficacy of this fuzzy neural system on larger data sets. The complementary nature of the neural network and the fuzzy logic lead us to believe that a further-refined fuzzy neural system will significantly improve the state-of-the-art pattern recognizers, especially in noisy environments.

ACKNOWLEDGMENTS

The author would like to thank the three anonymous referees for their constructive criticisms and helpful comments on the earlier version of this manuscript. This work was supported in part from Center for Artificial Intelligence Research (CAIR), the Engineering Research Center (ERC) of Excellence Program in Korea.

References

1. Hansen, L. K., and Salamon, P., Neural network ensembles, *IEEE Trans. Pattern Anal. Machine Intell.* 12, 993–1001, 1990.
2. Scofield, C., Kenton, L., and Chang, J., Multiple neural net architectures for character recognition, *Proceedings of Compcon*, San Francisco, IEEE Computer Soc. Press, 487–491, 1991.
3. Jacobs, R. A., Jordan, M. I., Nowlan, S. J., and Hinton, G. E., Adaptive mixtures of local experts, *Neural Comput.* 3, 79–87, 1991.

4. Wolpert, D., Stacked generalization, *Neural Networks* 5, 241–259, 1992.

5. Cho, S.-B., and Kim, J. H., Two design strategies of neural network for complex classification problems, *Proceedings of the 2nd International Conference on Fuzzy Logic & Neural Networks*, Iizuka, Japan, 759–762, 1992.

6. Perrone, M. P., and Cooper, L. N., When networks disagree: Ensemble methods for hybrid neural networks, in *Neural Networks for Speech and Image Processing* (R. J. Mammone, Ed.), Chapman-Hill, London, 1993.

7. Cho, S.-B., and Kim, J. H., A multiple network architecture combined by fuzzy integral, *Proceedings of the IEEE / INNS International Joint Conference on Neural Networks*, Nagoya, Japan, 1373–1376, 1993.

8. Cho, S.-B., and Kim, J. H., Combining multiple neural networks by fuzzy integral for robust classification, *IEEE Trans. Systems Man Cybernet.* 25(2), 380–384, 1995.

9. Yager, R. R., Element selection from a fuzzy subset using the fuzzy integral, *IEEE Trans. Systems Man Cybernet.* 23, 467–477, 1993.

10. Yager, R. R., On ordered weighted averaging aggregation operators in multi-criteria decisionmaking, *IEEE Trans. Systems Man Cybernet.* 18, 183–190, 1988.

11. Richard, M. D., and Lippmann, R. P., Neural network classifiers estimate Bayesian *a posteriori* probabilities, *Neural Comput.* 3, 461–483, 1991.

12. Shlien, S., Multiple binary decision tree classifiers, *Pattern Recognition* 23, 757–763, 1990.

13. Sugeno, M., Fuzzy measures and fuzzy integrals: A survey, in *Fuzzy Automata and Decision Processes*, North Holland, Amsterdam, 89–102, 1977.

14. Leszeynski, K., Penczek, P., and Grochulskki, W., Sugeno's fuzzy measures and fuzzy clustering, *Fuzzy Sets and Systems* 15, 147–158, 1985.

15. Tahani, H., and Keller, J. M., Information fusion in computer vision using the fuzzy integral, *IEEE Trans. Systems Man Cybernet.* 20, 733–741, 1990.

16. Freeman, H., Computer processing in line drawing images, *Comput. Survey* 6, 57–98, 1974.