*MVA '92* IAPR Workshop on Machine Vision Applications Dec. 7-9,1992, Tokyo

# Invariant Object Recognition Using Neural Network Ensemble on the CM

Daijin Kim and Minsoo Suk

Dept. of Computer Engineering, Dong-A University, Pusan, Korea

and

Dept. of Electrical and Computer Engineering, Syracuse University, Syracuse, N.Y., U.S.A

## ABSTRACT

This paper concerns machine recognition of objects from their images, where the recognition is invariant to scale, translation, and rotation. A neural network used for recognizing input objects is four layer backpropagation network and a cluster of interconnected units spanning four layers of each network forms a functional block called a column. The $90^{o}$ rotation invariance has been obtained by a specific interconnection scheme between the units in the first and second layers in the network. The scale invariance has been achieved by superimposing many columns of different spatial resolutions at a specific position on the input visual field. The translation invariance has been obtained by overlapping two adjacent columns of the same scale. The generalization ability of object recognition system has been improved by using neural network ensemble supported by a consensus voting scheme. The neural network ensemble has been implemented on the Connection Machine, in such a way that (i) all training samples are presented simultaneously, and (ii) multiple networks are implemented simultaneously. A set of rigorous experimentations using 2-D key images has been performed to demonstrate the usefulness of neural network ensemble for invariant object recognition in terms of computation time, convergence characteristics, rotation invariance, size invariance, translation invariance, and the effect of noise.

## 1. INTRODUCTION

We set the goals of this research as the development of an object recognition system which satisfies the following specific aims:

(1) It must provide size, translation, and rotation invariance. The invariance should be achieved from the system's structure itself, not from the system's training in all possible examples of different size, translation, and rotation.

(2) It must have good generalization ability. By generalization, we mean that the system should be able to correctly classify any new input patterns that were not included in the training sample set.

(3) The underlying computation of the system should be suitable for massively parallel implementation on a parallel machine such as the Connection Machine [1].

Our first step considered toward these goals was to determine an appropriate computational model. A neural network was chosen by some advantages that it could provide when used for pattern recognition such as (i) a good generalization ability (ii) massively parallel computation, and (iii) greater degree of robustness and fault tolerance by distributed processing. Among many neural networks, the multi-layer backpropagation network model [2] seems appropriate as the computational framework for implementing invariant object recognition systems since (i) the network constructs a complicated internal representation, (ii) the network has a good generalization ability, and (iii) backpropagation is the most effective current learning algorithm for the multi-layer neural network systems.

A neural network consisting of four feedforward layers was consequently developed. The network achieves rotation invariance from a particular interconnection scheme developed between the input layer and the second layer (receptive field layer). Each network can be considered as a functional column that recognizes an object with a specific size and a specific location. All columns have different spatial resolutions but identical weight vectors. The size invariance is achieved by superimposing many columns of different spatial resolutions; the translation invariance is obtained by overlapping many columns of the same resolution.

A neural network ensemble consisting of a set of neural networks was then developed to improve the generalization ability of the object recognition system. Each network works independently but in a cooperative manner such that each network makes a different decision for a given input, and the neural network ensemble determines a final decision from the collective decisions of all the networks. Different decisions by different networks represent different ways of generalizing a set of training samples. Fig. 1-1 shows a block diagram of the overall object recognition system.

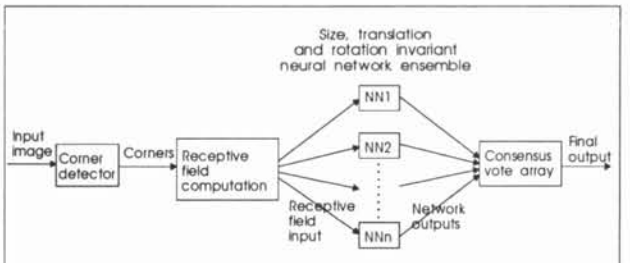



Fig. 1-1. A block diagram of the overall object recognition system.

The remainder of this paper describes the details of the neural network ensemble developed for object recognition, its implementation on the CM, and experimental results. Section 2 describes a network architecture for obtaining size, translation, and rotation invariant pattern classification. Section 3 describes a neural network ensemble which is used to improve the generalization performance of pattern classifier. Section 4 describes the implementation of the neural network ensemble onto the CM shortly and analyzes the experimental results and the generalization performance of the developed system. Finally, Section 5 draw a conclusion.

## 2. NN ARCHITECTURE FOR SCALE, TRANSLATION, AND ROTATION INVARIANT OBJECT RECOGNITION

As shown in Fig. 2-1, the basic element of our object recognition system is a four-layer feedforward neural network. A cluster of interconnected units spanning four layers of each network forms a functional block called a column. The input layer in a column consists of clusters of 49 input units. Each input unit has a localized receptive field of radius r with its center located at a particular point in the visual field. The input unit captures stimuli (i.e., corners) within the area of its receptive field.

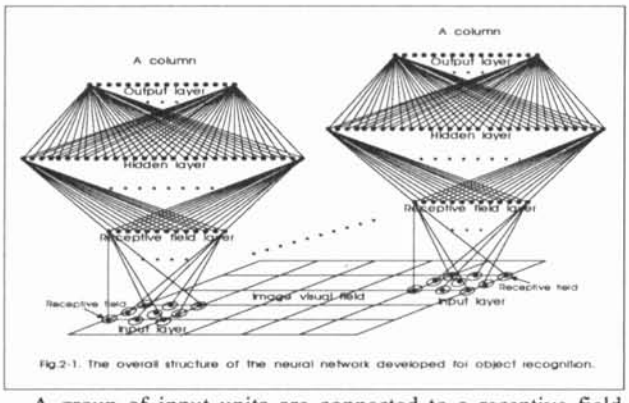


Fig.2-1. The overall structure of the neural network developed for object recognition.

A group of input units are connected to a receptive field unit in the second layer in order to obtain the rotation invariance such that (i) the angle difference between two adjacent input units is exactly $90^0$ and (ii) the input units are at the same distance away from the center of a column. (See 2-2-b) The only exception is the input unit at the center of a column, a unit that is directly connected to a receptive field. The 49 input units are locally connected to 13 receptive field (rf) units in the second layer (1 rf unit directly connected to the input unit at the center of a column + 12 rf units connected to other 48 input units). Fig. 2-2-a shows a typical interconnection scheme between receptive field units in the second layer and input units in the first layer. The detailed explanation how the $90^0$ rotation invariance is achieved can be found in [3]. The units in two adjacent layers from the second layer to the hidden layer (or from the hidden layer to the output layer) are fully connected. This structure completes one column of the overall network. The existence of the third layer is critical for some complicated applications for faster convergence [4].

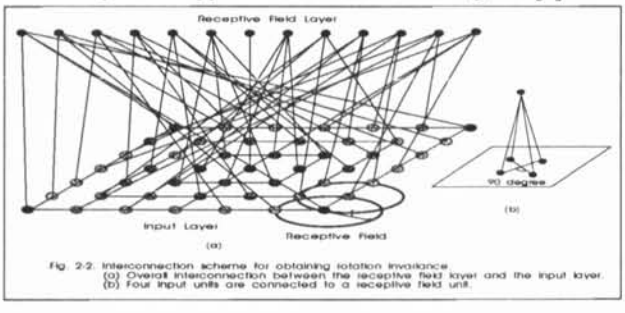


Fig. 2-2. Interconnection scheme for obtaining rotation invariance.
(a) Overall interconnection between the receptive field layer and the input layer.
(b) Four input units are connected to a receptive field unit.

In order to achieve scale and translation invariance, the object recognition system contains multiple columns of different resolutions. In the current implementation, there are 843 input units organized into three groups: 625 fine-scale input units (r = 8 pixels), 169 medium-scale input units (r = 16 pixels) and 49 large-scale input units (r = 32 pixels). Since two adjacent columns share 28 input units, there are 49 fine-scale columns (columns with fine-scale input units), 9 medium-scale columns, and 1 large scale-column.

In operation, all columns share an identical set of weights obtained from training only a single column. The overlap between two columns having identical weights vectors provides translation invariance. Translation of an object by less than one half of the width of a column can be tolerated since (i) the translation results in slight perturbation of its representation in the input layer of the column covering the object and (ii) the receptive fields in the input image field also overlap each other. Columns with different spatial resolutions provide scale invariance. An object scaled to a smaller size is recognized by the small-scale column while a bigger image of the same object is recognized by the large-scale column using the same weight vectors.

Now, we turn our attention to the learning algorithm for the developed network. The backpropagation learning procedure [5] can not be used directly for our network. The procedure has been modified for the links connecting the receptive field units in the second layer and the input units in the first layer. It is convenient to interpret the structure of our network in the following manner for understanding the modification:

(1) A receptive field unit in the second layer is considered as having four internal virtual units.

(2) It is assumed that the error, $\delta_r$, in the output of the receptive field unit, which is obtained by backpropagating errors from the hidden units, is equally distributed among four internal virtual units.

(3) The weight change between the receptive field unit and the input unit is the sum of the individual weight change of four internal virtual units.

Therefore, $\Delta w_{ij}$, the change in weights wij between the receptive field unit i and the input unit j can be written as

$$\Delta w_{ij} = \eta \frac{\delta_i}{4} \sum_{j=0}^{3} (1-o_j)\, o_j\, I_{(i-j) \bmod 4}$$

where $\delta_i$, $o_j$, and $I_j$ are the error of the receptive field unit i, the output value of the j-th internal state, and the value of the input unit j, respectively. The bias is treated as an input unit whose input value is always 1.0.

## 3. NN ENSEMBLE FOR OBJECT RECOGNITION

A neural network ensemble consists of a set of neural networks (called ensemble) running simultaneously. Each individual network in the ensemble has different weights due to different initial weights, different sequences of the training samples, or different partitionings of the input samples used during the training. The different weights correspond to different generalizations of the same set of training samples. The decision made by each individual network may or may not be a correct decision. The final decision made by the ensemble is based on the all decisions made by the individual network. We shall show that the collective decision is much more accurate because different networks make generalization error on different subsets of the input space and because the union of these subsets makes the area of input space covered by a neural network ensemble larger.

Fig. 3-1 illustrates how the generalization is improved by a neural network ensemble. In this figure, U and T constitutes a universe of possible input-output pairs as well as a training sample set, respectively. We assume that k networks are trained by T independently. Then, the networks produce k different weight vectors, $\mathbf{w}_1,\mathbf{w}_2,\ldots,\mathbf{w}_k$. These vectors respond differently to a new input. In other words, the weight vectors make different generalizations in the input space, $G_1,G_2,\ldots,G_k$, respectively. As shown in Fig. 3-1, the area in the input space covered by a neural network ensemble is expanded gradually. Therefore, a certain input misclassified by a network can be classified correctly by another network, which implies the improvement of generalization performance.

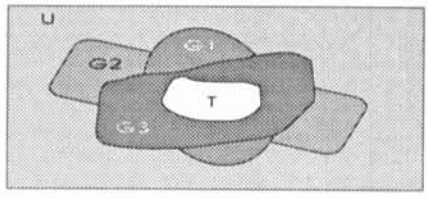


Fig. 3-1. Generalization performance is improved by a neural network ensemble.

There are several different ways to arrive at a final decision from the decisions made by individually trained networks. The most commonly used decision rules are the plurality-voting rule and the majority-voting rule [6]. The plurality-voting rule chooses the output that is agreed upon by more networks than any other. The majority-voting rule takes an output that is agreed upon by more than half of the networks. When there is no such agreement, the result is considered an error. In our work, two decision rules are used in sequence: (i) first, the majority rule is used to decide an output of neural network ensembles and (ii) if the rule (i) fails to reach an output, the plurality decision rule is applied.

## 4. EXPERIMENTAL RESULTS AND DISCUSSIONS

### 4.1. Feature Extraction

We choose corner points of the boundary of an object as features since (i) they are viewpoint invariant, (ii) they reduce the amount of data to be processed, while at the same time, preserving important information about the object, and (iii) their extraction can be done by local computation, thus make it suitable for parallel processing. For our research, we adapt the Medioni and Yasumoto algorithm [7] since (i) the use of cubic B-spline eliminates unusual noisy corners by smoothing digital curve contours, and (ii) the computation is local, which is suitable for implementing on a parallel machine such as the CM. The overall feature extraction process is illustrated in Fig. 4-1.

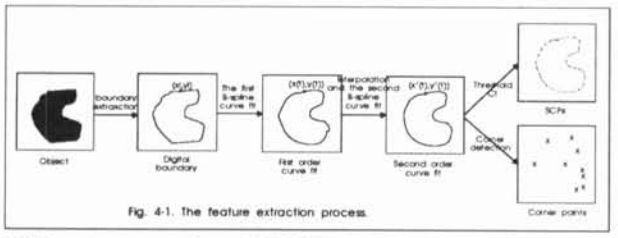


Fig. 4-1. The feature extraction process.

### 4.2. Implementation of NN Ensemble on the CM

The implementation presented here is organized into three different levels. A neural network ensemble (level 3) consists of N network groups. Each network group (level 2) consists of n identical networks. A network (level 1) consists of m identical columns, where each column is mapped to a processor of the CM. Each column consists of (i) four units (one unit per one layer) and (ii) weight links connected to the units in the column. Fig. 4-2 shows a schematic diagram of three levels. They are integrated in a cubical geometry on the CM as shown in Fig. 4-3. The geometry denoted by g(N,n,m), where N, n, and M is the number of network groups, the number of networks, the size of network, respectively, requires $N \cdot n \cdot m$ virtual processors. Each processor allocates $4 + 3 \cdot m$ local memory fields in the case of fully-connected networks.

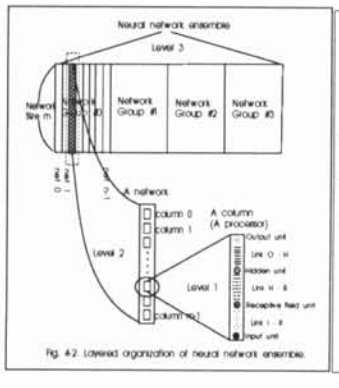


Fig. 4-2. Layered organization of neural network ensemble.

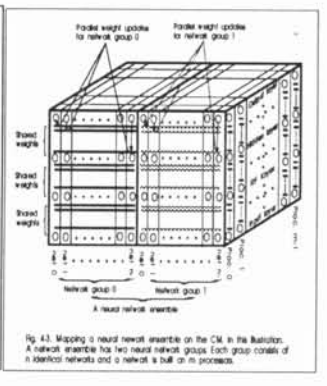


Fig. 4-3. Mapping a neural network ensemble on the CM. In this illustration, A network ensemble has two neural network groups. Each group consists of n identical networks and a network is built on m processors.

The detailed description of implementation of neural network ensemble on the CM is found in [3]. The implementation of neural network ensemble has the follwoing features:

1. All training samples are presented to a network group simultaneously, one training sample per one network.

2. Every network group has the same input-output training samples. Thus, the same set of training samples is copied to all network groups. All network groups are trained simultaneously with different initial weight constants. Initial weight constants obtained from the internal random number generator of the CM are stored across the processors sharing the same set of data to save the required memory space.

3. The final output of a network group of individually trained networks is selected as follows: First, the output unit having maximum value in each network is chosen, and then the winner among these output units is selected as a final output of the network group.

### 4.3. Description of the Experiment

In the experiments, we classified an input image into one of 16 different classes, corresponding to 16 different types of keys. Six of the types had very similar shapes. The neural network ensemble used in the experiments consisted of 16 network groups having different weight vectors. Each network group consisted of 96 networks whose weight constants were shared among them. Each network had four layers -- the first consisting of 52 input units, the second consisting of 13 receptive field units, the third consisting of 32 hidden units, and the fourth consisting of 16 output units.

The images were obtained using the HP Scanjet Scanner (150 dpi) and stored into PCX format. Each image of 64 x 64 pixels contained an object of about 54 x 54 pixels. Each key was laid so that its major axis was aligned with the vertical direction of the image field. We extracted corner points from the boundary contours of the input images. Fig 4-4 shows the examples of boundary contours of input images along with the corner points marked by x.

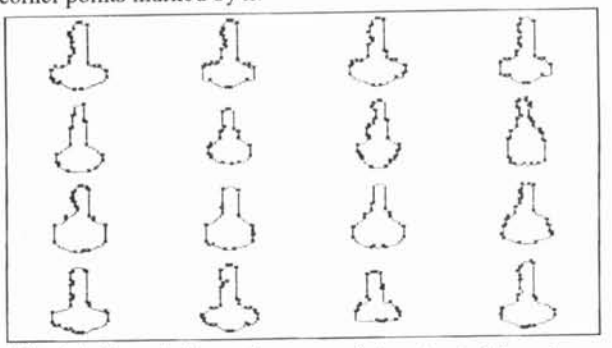


Fig. 4-4. Boundaries and corner points extracted from keys

### 4.4. Performance Analysis

#### 4.4.1. Rotation Invariance

First, we tested the rotation invariance property. The experimental result showed a small misclassification rate for the test samples rotated through multiples of $15^{o}$. The misclassification rates among 16 networks ranged from 0.9% to 7.5%. Fig. 4-5 shows the error probabilities for the samples rotated by other angles ($15 \cdot i < \alpha < 15 \cdot (i+1)$, where i=0,1,...,23). Each point on the curves was obtained by averaging error rates made by 384 test samples (16 keys x 24 rotations ($15^{o} \cdot i + \Delta\theta$, where i= 0,1,...,23 and $\Delta\theta$ is one of $2.5 \cdot j$, where j=0,1,..,5 )). From the Fig. 4-5, we note that

(1) As expected, the error rates were highest for test samples rotated by $15^{o} \cdot i + 7.5$, where i= 0,1,...,23.

(2) The average error rate of all individual networks over all rotation angles was 9.1%, but the error rate by the neural network ensemble (NN=15) using plurality- (or majority-) voting over all rotation angles was only 1.5% (or 1.9%). This result demonstrates that the neural network ensemble increases generalization ability significantly.

Fig. 4-5. Comparison of different decision methods of NN Ensemble for Rotated Objects

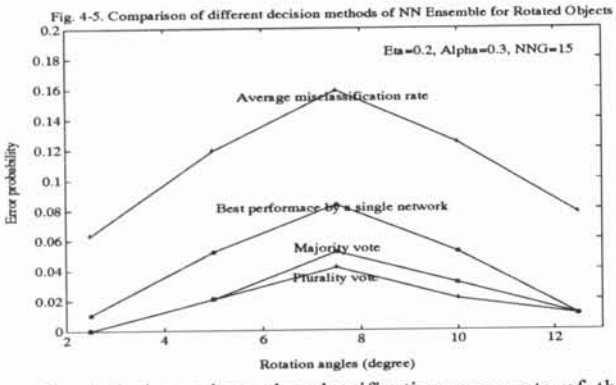
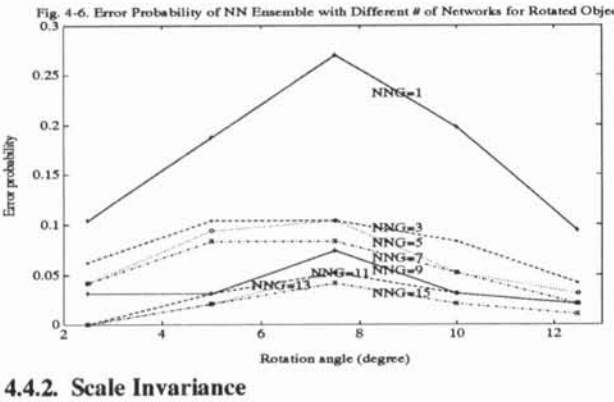



Fig. 4-6 shows how the classification error rate of the neural network ensemble was changed with the number of networks using the plurality-voting method. When a single network classified test samples, the error rate was high (in the worst case, 27 %). As more networks were used, the error rate decreased over all rotation angles. The decrease around the rotation of $7.5^{o}$, which was the worst case, was quite remarkable. Also, note that the gain achieved by using more than 11 networks was insignificant.

Fig. 4-6. Error Probability of NN Ensemble with Different # of Networks for Rotated Objects




### 4.4.2. Scale Invariance

Next, we tested the scale invariance of the neural network ensemble. Fig. 4-7 shows the structure of the neural network ensemble for testing the size invariance. An image field consisted of 8 different columns whose input sizes were $0.25 \cdot S \cdot i$, where S was the normal size of a column, and i = 1,2,...,8. The largest column had a 128 x 128 pixel array. Eight different column inputs of different spatial resolutions were applied to 16 different networks. 128 different decisions made by 16 networks were sent to the vote array for making a final decision. The vote array had 16 x 8 elements, and each element consisted of an object index and an output value. An example of the output of vote array is shown in Table 4-1, where the element at the intersection of the row $NN_i$ and column $S_j$ is the decision output of i-th network about j-th column input whose size is $0.25 \cdot S \cdot j$. The true object index of the test sample used for this example was 1.

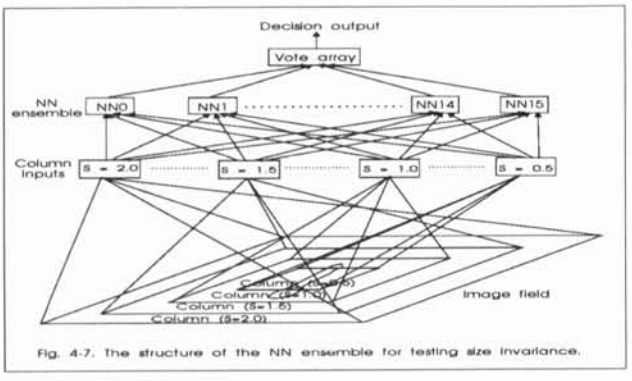



Fig. 4-7. The structure of the NN ensemble for testing size invariance.

In determining the final output, first, the object index which had maximum votes in each column in the vote array was determined. Second, we chose as the final output the object index having maximum votes among the object indexes of all columns, which had been chosen in the first step. When a tie occurred among those indexes, we selected the one which had the largest sum of output values. Using the decision rule explained above, we made the following observations from Table 4-1:

(1) When a single network was used, the network often made a wrong decision. For example, when $NN_0$ network was used alone, the test sample was classified as object 9 rather than as the true object 1 because the $S_0$ column had the maximum output value.

(2) When seven networks ($NN_0$ - $NN_6$) were used, the ensemble would classify the test sample as object 4 rather than the true object 1 because the sum of output values (=6.661) in the $S_1$ column is larger than the sum of output values (=6.606) in the $S_2$ column although both decisions have tie votes (=7).

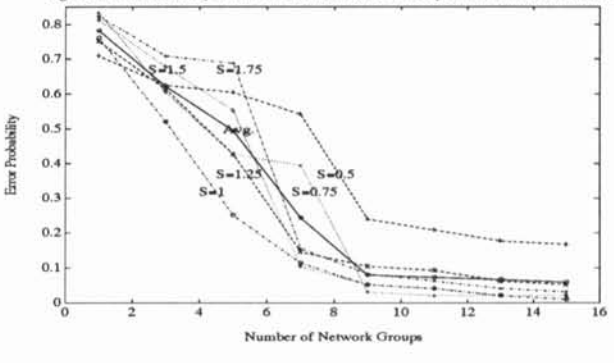
(3) When N > 8, the network ensemble always made the right decision because all networks classified the test sample as object 1 in the $S_2$ column.

| | $S_0$ | $S_1$ | $S_2$ | $S_3$ | $S_4$ | $S_5$ | $S_6$ | $S_7$ |
|---|---|---|---|---|---|---|---|---|
| $NN_0$ | (9,0.99) | (4,0.98) | (1,0.95) | (1,0.96) | (9,0.97) | (9,0.92) | (10,0.91) | (10,0.94) |
| $NN_1$ | (1,0.94) | (4,0.94) | (1,0.96) | (5,0.94) | (5,0.97) | (5,0.95) | (5,0.97) | (5,0.91) |
| $NN_2$ | (10,0.97) | (4,0.99) | (1,0.95) | (1,0.96) | (1,0.94) | (7,0.78) | (7,0.79) | (10,0.49) |
| $NN_3$ | (10,0.68) | (4,0.97) | (1,0.97) | (1,1.0) | (1,0.65) | (1,0.26) | (8,0.20) | (8,0.40) |
| $NN_4$ | (10,0.97) | (4,0.99) | (1,0.97) | (5,0.92) | (10,0.74) | (10,0.84) | (10,0.84) | (10,0.85) |
| $NN_5$ | (10,0.94) | (4,0.96) | (1,0.85) | (1,1.0) | (5,0.69) | (2,0.90) | (2,0.93) | (2,0.92) |
| $NN_6$ | (9,0.89) | (4,0.83) | (1,0.95) | (5,0.92) | (5,0.84) | (10,0.95) | (10,0.98) | (10,0.99) |
| $NN_7$ | (10,0.98) | (15,0.80) | (1,0.91) | (1,0.26) | (1,0.15) | (7,0.13) | (7,0.15) | (7,0.16) |
| $NN_8$ | (9,0.51) | (4,0.96) | (1,0.95) | (5,0.98) | (5,0.84) | (5,0.85) | (5,0.82) | (5,0.77) |
| $NN_9$ | (9,0.45) | (4,0.98) | (1,0.95) | (2,0.84) | (2,1.0) | (2,0.99) | (2,0.94) | (10,0.79) |
| $NN_{10}$ | (10,0.92) | (4,1.0) | (1,0.97) | (6,0.94) | (1,0.97) | (5,0.91) | (5,0.94) | (5,0.94) |
| $NN_{11}$ | (4,0.14) | (4,0.90) | (1,0.90) | (1,0.99) | (2,1.0) | (2,1.0) | (2,1.0) | (10,0.98) |
| $NN_{12}$ | (10,0.96) | (14,0.46) | (1,0.87) | (2,0.98) | (2,0.99) | (2,1.0) | (2,1.0) | (2,1.0) |
| $NN_{13}$ | (10,0.93) | (4,0.86) | (1,0.93) | (1,0.06) | (8,0.96) | (8,0.95) | (8,0.87) | (8,0.78) |
| $NN_{14}$ | (9,0.72) | (15,0.95) | (1,0.95) | (2,0.87) | (14,0.76) | (2,0.74) | (5,0.85) | (5,0.6) |
| $NN_{15}$ | (9,0.99) | (9,0.59) | (1,0.94) | (2,0.68) | (2,1.0) | (2,1.0) | (2,1.0) | (2,0.99) |

Table 4-1. A typical vote array for deciding an final output from multiple networks.

Fig. 4-8 shows the error probability of the neural network ensemble for the five different scale factors, 0.75, 0.8125, 0.875, 0.9075, and 1.0. In the cases of the scale factors of 0.75 or 1.0, the improvement made by using the neural network ensemble was significant. This improvement was expected since there existed the columns whose scaling factors were S=0.75 and S=1.0, respectively. Also, as expected, the improvement was least for the scale factor in the middle of 0.75 and 1.0.

Fig. 4-8. Error Probability of the NN Ensemble for Scaled Objects vs Number of Networks



### 4.4.3. Translation Invariance

Next, we tested the translation invariance of the neural network ensemble. For the simplicity of discussion, we assume that the translation occurs along the x-axis. As explained in the Section 2, translation invariance was obtained by overlapping two adjacent columns. We placed 8 additional columns between two adjacent columns whose centers were separated by 8·T pixels, where T = ±4,±3,±2, and ±1, from the center of a specific column (see Fig. 4-9). 640 translated test samples were generated (16 keys x 8 translations x 5 variations/translation) where the amount of variations was chosen randomly between -2 and 2. The final output from the vote array was determined by consensus-voting as in the testing of scale invariance.

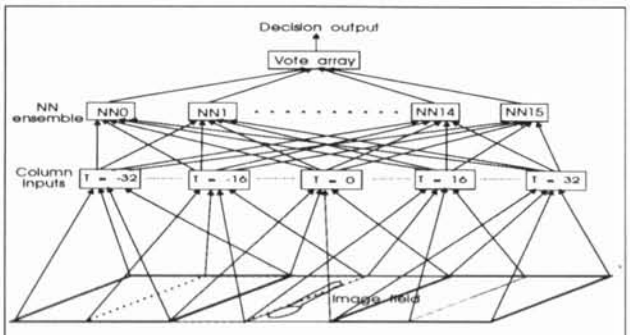
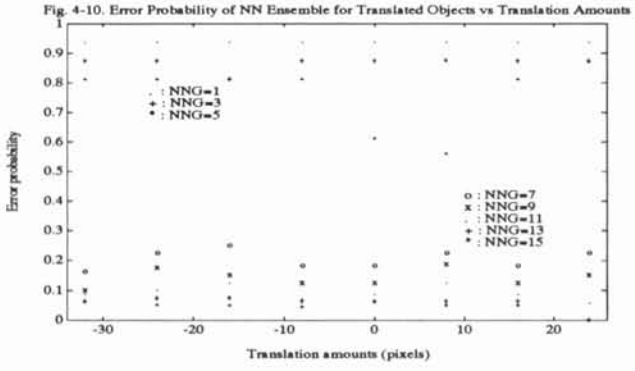



Fig. 4-9. The structure of the NN ensemble for testing translation invariance.

Fig. 4-10 shows the performance of the neural network ensemble for the translated test images. From the figure, we noted that the recognition by a single network produced a large error rate ( > 0.9) because each individual network was trained only by a typical pose of each object. Therefore, the decision output of the column whose position was nearest to the input image was not always maximum among the 9 different output values of the different columns (see Table 4-1). As the size of the neural network ensemble increased, the error rate decreased significantly (when NN > 5). This decrease occurred because the decisions made by the networks in the ensemble were likely to make a strong consensus gradually as the number of networks increased. This unusual characteristic of the neural network ensemble could also be observed in the case of size invariance.




## 5. CONCLUSION

The main contribution of the research was the development of a neural network ensemble which could be used for recognizing 2-D objects invariant to their size, position, and orientation. The scheme was also robust to additive noise. In the test of rotation invariance, the average error rate obtained by a single network was about 10%, which dropped to 1.5% when 15 networks were used. The improvement made by the neural network ensemble was more apparent for size invariance and translation invariance. For scaled images and translated images, the classification accuracy of a single network was totally unacceptable, the classification error rate reaching higher than 90%. The error rate dropped to 5 % when the 15 network ensemble was used with a consensus voting array. The noise immunity of the neural network ensemble was also tested. The misclassifcation error rate at 20% additive noise was dropped from the 48%, using a single network, to 10%, using an ensemble of 15 networks.

Another contribution of the research was the implementation of a parallel simulation tool for neural network ensembles on the CM. The simulation program could be used for any feedforward backpropagation neural networks of different sizes, and any number of networks within the limit of the CM's memory allocation. The simulation of the neural network ensemble on the CM produced the speed of 92 MIPS. This speed enabled us to perform the trainings of the 16 network ensemble within the 2 hours. The modified backpropagation learning also accelerated the early phase of the training at the cost of the instability of the system.

In the future, it is hoped that the system described here can be extended toward 3-D object recognition. By using the edges as 2-D image features of 3-D objects and training the network with the nonaccidental properties of the image features and with the geometrical relationships of geons [8] within a 3-D object, the network described here could be used for 3-D object recognition without a major modification.

## REFERENCES


[1] Thinking Machine Corporation, "Connection machine model CM-2 technical summary," *Technical Report*, No. 86-14, Cambridge, MA, 1986.

[2] D. Rumelhart, G. Hinton, and J. MaClelland, "Learning internal representations by error propagation," *Parallel Distributed Processing: Explorations in the Microstructure of Cognition,vol. 1: Foundation*, pp. 318-362, Cambridge MA, Bradford Book/MIT Press, 1986.

[3] Daijin Kim, "Invariant Object Recognition Using Neural Network Ensemble on the Connection Machine," *Ph.D Dissertation*, Syracuse University, 1991

[4] J. K. Kruschke, "Creating local and distributed bottlenecks in hidden layers of back-propagation networks," *Proceedings of the 1988 Connectionist Models*, pp. 120-126, June, 1988.

[5] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representation by back-propagation errors," *Nature*, vol. 323, pp. 533-536, 1986.

[6] L. K. Hansen and P. Salamon, "Neural network ensembles", *IEEE Trans. Pattern Anal. Machine Intell.*,vol. PAMI-12, no. 10, pp. 993-1001, Oct. 1990.

[7] G. Medioni and Y. Yasumoto, "Corner detection and curve representation using cubic B-splines," *Comput. Graph. Image Processing*, vol. 39, pp. 267-278, 1987.

[8] I. Biederman, "Recognition-by-components: A theory of human image understanding," *Psychological Review*, vol. 94, no.2, pp. 115-147, 1987.