

# SHAPE RECOGNITION WITH NEAREST NEIGHBOR ISOMORPHIC NETWORK

Hung-Chun Yau      Michael T. Manry  
Department of Electrical Engineering  
University of Texas at Arlington  
Arlington, Texas 76019

**Abstract - The nearest neighbor isomorphic network paradigm is a combination of sigma-pi units in the hidden layer and product units in the output layer. Good initial weights can be found through clustering of the input training vectors, and the network can be successfully trained via back propagation learning. We show theoretical conditions under which the product operation can replace the Min operation. Advantages to the product operation are summarized. Under some sufficient conditions, the product operation yields the same classification result as the Min operation. We apply our algorithm to a geometric shape recognition problem and compare the performances with those of two other well-known algorithms.**

## INTRODUCTION

As pointed out by Duda and Hart [1] and Fukunaga [2], the nearest neighbor classifier (NNC) approximates the minimum error Bayesian classifier in the limit as the number of reference vectors gets large. When the feature vector joint probability density is unknown, the NNC would be the preferred classifier, except for two problems. First, a prohibitive amount of computation is required for its use. Second, the NNC's performance is usually not optimized with respect to the training data.

As more hardware for parallel processing becomes available, the first problem will be solved. Several neural networks which are isomorphic to NNC's have been developed to attack the second problem. These include the learning vector quantization (LVQ) of Kohonen [3], the counter-propagation network of Hecht-Nielsen [4], the adaptive-clustering network of Barnard and Casasent [5], and the nearest neighbor isomorphic network (NNIN) of Yau and Manry [6]. In this paper we discuss properties of product units that allow them to substitute for Min units, in the NNIN. We compare its performance to that of LVQ2.1 for the geometric shape recognition problem.

## NETWORK TOPOLOGY AND TRAINING

The NNIN, which is a back propagation (BP) network isomorphic to a type of NNC, uses fairly conventional sigma-pi units [7] in the hidden layer and units similar to the product units [8] in the output layer. A set of good initial weights and thresholds can be directly determined from the reference feature vectors via appropriate mapping equations.

Each pattern is represented by a dimension- $N_f$  feature vector,  $\mathbf{x} = [x(1) \ x(2) \ \dots \ x(N_f)]^T$ . As shown in Fig. 1, the first processing layer consists of  $N_c \cdot N_s$  sigma-pi units which are connected to the  $N_f$  input features. Here  $N_c$  is the number of classes and  $N_s$  is the number of clusters per class. Let  $Snet(i,j)$ ,  $S\theta(i,j)$ , and  $Sout(i,j)$  denote the net input, threshold, and output of the  $j$ th unit of the  $i$ th class, respectively.  $Sw1(i,j,k)$  and  $Sw2(i,j,k)$  denote the connection weights from the  $k$ th input feature to the  $j$ th sigma-pi unit of the  $i$ th class. The sigma-pi unit net and activation are respectively

$$Snet(i,j) = S\theta(i,j) + \sum_{k=1}^{N_f} [Sw1(i,j,k) x(k) + Sw2(i,j,k) x^2(k)]$$

$$Sout(i,j) = \frac{1}{1 + \exp[-Snet(i,j)]}$$

The second processing layer is composed of product units [8] with each feature raised to the first power. Let  $P\theta(i)$ ,  $Pnet(i)$  and  $Pout(i)$  denote the threshold, net input and output of the  $i$ th unit in the second layer. The  $Pw(i,\hat{i})$  are the connection weights for  $Pin(i)$  and  $Pnet(i)$ .

$$Pin(i) = \prod_{j=1}^{N_s} Sout(i,j), \quad Pnet(i) = P\theta(i) + \sum_{\hat{i}=1}^{N_c} Pw(i,\hat{i}) Pin(i),$$

$$Pout(i) = \frac{1}{1 + \exp[-Pnet(i)]}$$

The NNIN can be initialized and trained as follows. Let  $\mathbf{r}_{ij} = [r_{ij}(1), r_{ij}(2), \dots, r_{ij}(N_f)]^T$  and  $\mathbf{v}_{ij} = [v_{ij}(1), v_{ij}(2), \dots, v_{ij}(N_f)]^T$  respectively be the mean and variance vectors of the  $j$ th cluster of the  $i$ th class. Define the squared distance of the vector  $\mathbf{x}$  to  $\mathbf{r}_{ij}$  as

$$D_{ij}^2 = \sum_{k=1}^{N_g} \frac{[x(k) - r_{ij}(k)]^2}{v_{ij}(k)}$$

Comparing  $Snet(i,j)$  with  $D_{ij}^2$ , we may assign the initial weights and threshold of the  $j$ th sigma-pi unit of the  $i$ th class as

It is simple to initialize the second layer as  $Pw(i,\hat{i}) = \delta(i-\hat{i})$  and  $P\theta(i) = 0$ .

Let  $Tout(i)$  be the desired output.  $N_p$  denotes the total number of training

$$S\theta(i,j) = \sum_{k=1}^{N_f} \frac{r_{ij}^2(k)}{v_{ij}(k)}, \quad Sw1(i,j,k) = \frac{-2r_{ij}(k)}{v_{ij}(k)}, \quad Sw2(i,j,k) = \frac{1}{v_{ij}(k)}$$

patterns. Using the  $q$ -norm, which is the  $p$ -norm of [9], the system performance measure is defined as

$$E_T = \sum_{p=1}^{N_p} \left( \sum_{i=1}^{N_c} [T_{out}(i) - P_{out}(i)]^{2q} \right)^{1/q}$$

where  $q$  is a positive integer. In practice we alter the error criterion from the least square approach ( $q=1$ ) to the minimax approach ( $q=\text{infinity}$ ) or vice versa when the learning process slows. In our experiments, this adaptive- $q$  technique results in an increase in learning speed.

## APPROPRIATENESS OF THE PRODUCT UNIT

In a classical NNC implementation, Min units would be used in the output layer to determine the classification result. A Min unit passes the minimum value of its inputs through to its output. In the NNIN, it is quite possible to use Min units in the output layer. This would result in a network very similar to those of Kohonen. In this section, we consider alternative units. In the NNC, the Min units have two important properties that allow the generation of complicated (disjoint for example) decision regions. These are as follows.

**Property 1:** Let  $\mathbf{r}_{ij}$  be the vector among the  $\mathbf{r}_{ij}$  from which the random vector  $\mathbf{x}$  has the smallest distance  $D_{ij}$ . If  $\mathbf{x} \rightarrow \mathbf{r}_{ij}$ , then  $D_{ij} \rightarrow 0$ , and  $D_{ij} = \min D_{ij}$  for all  $i \in [1, N_c], j \in [1, N_s]$ .

**Property 2:** Property 1 still holds if new reference vectors are added to any class.

Two possible replacements for the Min unit are the product unit and the sum unit, whose activations are respectively

$$Pin(i) = \prod_{j=1}^{N_s} D_{ij}^2, \quad S(i) = \sum_{j=1}^{N_s} D_{ij}^2 \quad (1)$$

The product unit has both of the properties given above. As  $\mathbf{x} \rightarrow \mathbf{r}_{ij}$ , then  $D_{ij}$  and  $Pin(i)$  both approach 0. Assuming all the  $\mathbf{r}_{ij}$  vectors are unequal, then  $D_{ij} > 0$  and  $Pin(i) > 0$  for all  $i \neq \hat{i}$  and Property 1 is satisfied. When a new reference vector is added, Property 1 and therefore Property 2, are still satisfied. Therefore the product units have the required properties and can function as Min units.

For the sum unit activation of (1), as  $\mathbf{x} \rightarrow \mathbf{r}_{ij}$ ,  $D_{ij}$  approaches 0 but  $S(i)$  does not. Property 1 is not generally satisfied for the sum unit. If a sum unit did have Property 1 and if a new example vector is added to the  $\hat{i}$ th class,  $S(\hat{i})$  can no longer be driven to 0 as  $\mathbf{x}$  approaches  $\mathbf{r}_{ij}$ . Thus the sum unit does not have

properties 1 or 2, and is not a suitable replacement for the Min unit.

## COMPARISONS OF THE PRODUCT AND MIN UNITS

The two advantages to using product units rather than Min units are that (a) derivatives of products are much simpler to calculate than derivatives of a minimum operation, and (b) a back-propagation iteration for product units results in changes for all weights, unlike a similar iteration for min units.

Assume that  $P_w(i, \hat{i}) = \delta(i - \hat{i})$ . For each class, put the cluster outputs  $Sout(i, j)$  in increasing order such that  $Sout(i, 1) \leq Sout(i, 2) \leq \dots \leq Sout(i, N_s)$ . Assume that the correct class is class 1. The Min unit assigns the vector  $\mathbf{x}$  to class 1 if

$$Sout(1, 1) < Sout(i, 1) \quad \text{for } i > 1 \quad (2)$$

The product unit assigns  $\mathbf{x}$  to class 1 if  $Pin(1)$  is smaller than  $Pin(i)$  for every  $i > 1$ , which is the same condition as

$$\begin{aligned} Sout(1, 1) < Sout(i, 1) \left( \prod_{j=2}^{N_s} Sout(i, j) / \prod_{j=2}^{N_s} Sout(1, j) \right) \\ = Sout(i, 1) \cdot Rs(i) \end{aligned} \quad (3)$$

for all  $i > 1$ . If (2) is true, (3) is true if either of the following sufficient conditions are true: (a)  $Sout(1, j) < Sout(i, j)$  for all  $i > 1$  and  $j \geq 2$ ; and (b)  $Rs(i) < Rs(1)$  for all  $i > 1$ . Also, if (8) is true, either of the above conditions is sufficient for (7) to be true. Thus under certain conditions, product units give the same answer as Min units.

## SHAPE RECOGNITION EXPERIMENT

In this section, we compare our algorithm's performance to that of LVQ2.1 and the NNC for a geometric shape recognition problem. The four primary geometric shapes used in this experiment are ellipse, triangle, quadrilateral, and pentagon. Each shape image is binary-valued with  $64 \times 64$  pixels. There were 200 training patterns and 200 testing patterns, each with varying amounts of rotation, scaling, translation, and oblique distortions. Example shapes are shown in Fig. 2. The feature sets used include circular harmonic expansion (CHE) features, log-polar transform (LPT) features, and ring-wedge energy features (RWE). Detailed feature definitions are given in the appendices. The feature vectors were of dimension 16. All three classifiers were initialized with 5 reference clusters per class through the K-mean clustering algorithm [10].

In figures 3-5, we plot the recognition error percentage versus the iteration number for training. The classifier performances are summarized in Table 1. In this experiment, the NNIN generally has better training speed and classification performance than the NNC and LVQ2.1.

**TABLE 1. ERROR PERCENTAGE IN TRAINING/TESTING**

Classifier	CHE	LPT	RWE
NNC	3.38/6.88	4.38/5.13	5.38/8.25
LVQ2.1	1.50/2.88	2.13/4.30	3.00/7.13
NNIN	0.13/2.75	0.00/1.75	0.38/6.13

## CONCLUSIONS

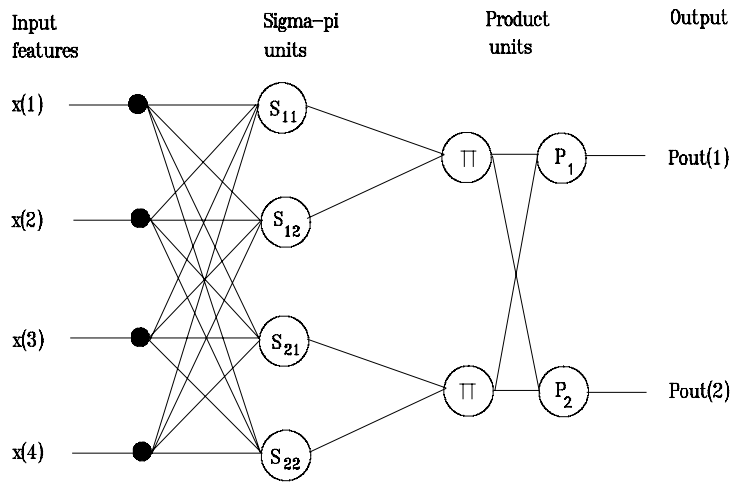
In this paper we have analyzed the properties of the product unit, and shown that it is a valid replacement for the Min unit in the NNIN. We have demonstrated that good initial weights enhance learning in the NNIN. It has been shown that the NNIN can be trained efficiently using the gradient descent backpropagation algorithm. The NNIN is found to perform better and learn faster than the NNC and LVQ2.1 for the particular feature sets and recognition problems tried. Since the classification performance of the network is improved for both training and testing data, it is apparent that the classifier is generalizing, and not just memorizing the training data.

Acknowledgement; This work was funded by the State of Texas through the Advanced Technology Program.

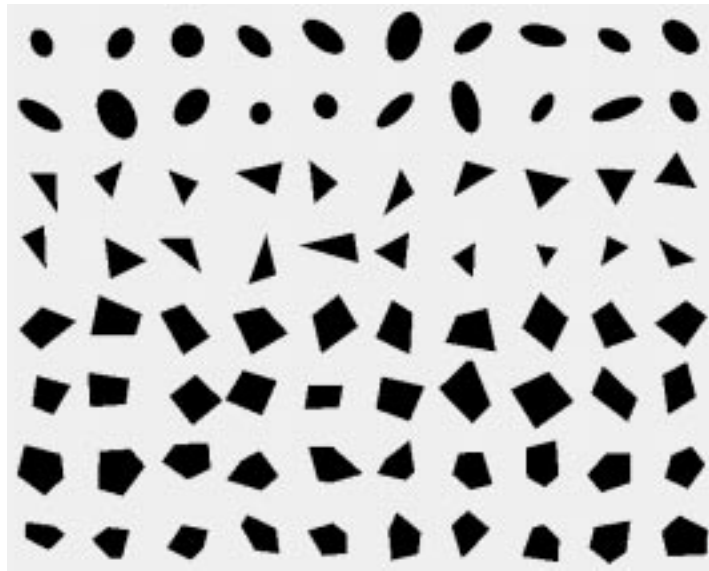
## REFERENCES

- [1] R.O. Duda and P.E. Hart, Pattern Classification and Scene Analysis, Wiley, New York, 1973.
- [2] K. Fukunaga, Introduction to Statistical Pattern Recognition, Academic Press, New York, 1972.
- [3] T. Kohonen, "Improved versions of learning vector quantization", IJCNN, San Diego, Ca., Vol.1, 1990, pp.545-550.
- [4] R. Hecht-Nielsen, "Applications of counterpropagation network", Neural Networks, Vol.1, 1988, pp.131-139.
- [5] E. Barnard, and D. Casasent, "Image processing for image understanding with neural network", IJCNN, Washington, D.C., Vol.1, 1989, pp.111-115.

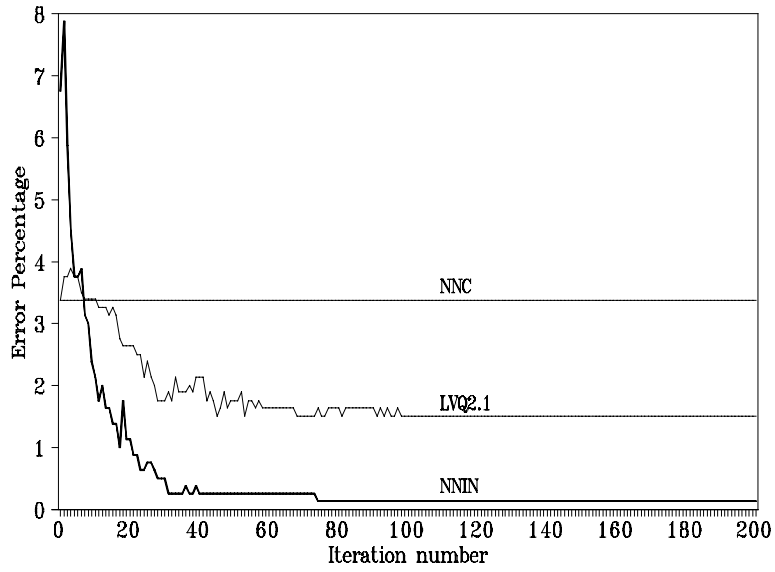
- [6] H.C. Yau and M.T. Manry, "Sigma-pi implementation of a nearest neighbor classifier", IJCNN, San Diego, Ca., Vol.1, 1990, pp.667-672.
- [7] D.E. Rumelhart and J.L. McClelland (Eds.), Parallel Distributed Processing, Vol.1, The MIT Press, Cambridge, Massachusetts, 1986.
- [8] R. Durbin and D.E. Rumelhart, "Product units: a computationally powerful and biologically plausible extension to backpropagation networks", Neural Computation, Vol.1, 1989, pp.133-142.
- [9] E.W. Cheney, Introduction to Approximation Theory, Chelsea Publishing, New York, 1982.
- [10] M. Anderberg, Cluster Analysis for Applications, Academic Press, New York, 1973.
- [11] Y-N Hsu and H.H. Arsenault, "Pattern discrimination by multiple circular harmonic components", Applied Optics, Vol.23, 1984, pp.841-844.
- [12] D. Psaltis and D. Casasent, "Deformation invariant optical processors using coordinate transformations", Applied Optics, Vol.16, 1977.
- [13] J.L. Smith and R.D. Douglas, "Hybrid optical/electronic pattern recognition with both coherent and noncoherent operations", SPIE Vol.938, 1988, pp.170-177.
- [14] N. George, S. Wang and D.L. Venable, "Pattern recognition using the ring-wedge detector and neural-network software", SPIE Vol.1134, 1989, pp.96-106.
- [15] R. Wu and H. Stark, "Rotation-invariant pattern recognition using a vector reference", Applied Optics, Vol. 23, 1984, pp.838-840.



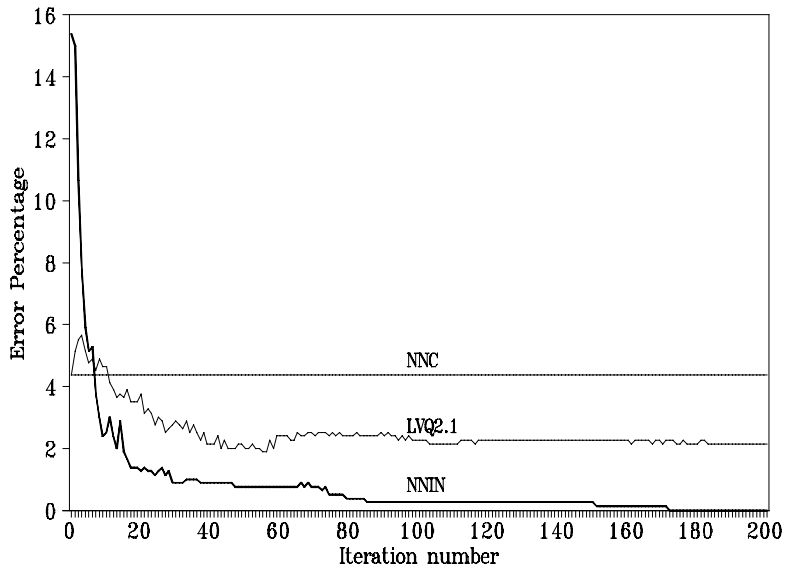
**Figure 1.** Topology of Nearest Neighbor Isomorphic Network



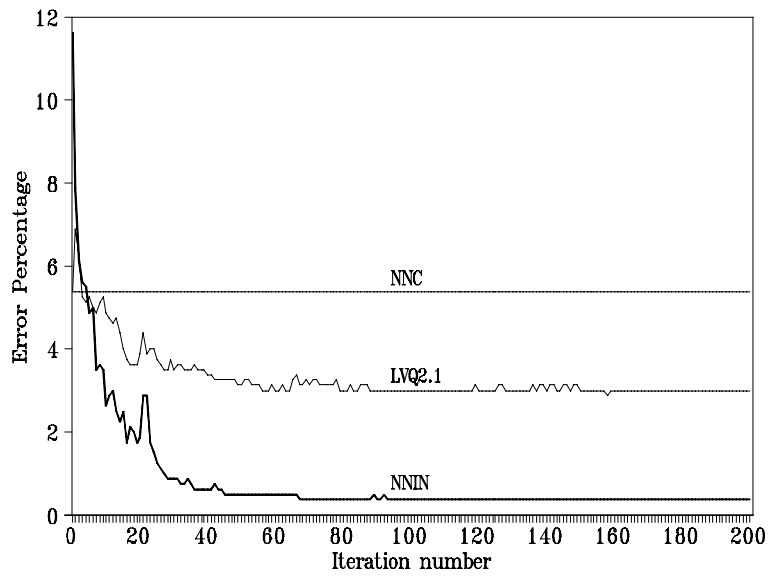
**Figure 2.** Typical geometric shapes



**Figure 3.** Neural networks in training for CHE feature set



**Figure 4.** Neural networks in training for LPT feature set



**Figure 5.** Neural networks in training for RWE feature set

## APPENDIX A. CIRCULAR HARMONIC EXPANSION FEATURES

Hsu and Arsenaul [11] have used the circular harmonic expansion (CHE) in the design of optical pattern recognition systems. After choosing the origin in polar coordinates, the circular harmonic expansion of the image,  $f(r, \theta)$  can be represented as the summation of the circular harmonics,

$$f(r, \theta) = \sum_{m=-\infty}^{\infty} f_m(r) e^{jm\theta}, \quad f_m(r) = \frac{1}{2\pi} \int_0^{2\pi} f(r, \theta) e^{-jm\theta} d\theta$$

We use the image's centroid as the origin in the expansion. The CHE features are defined as the energy of the circular harmonics over radius by

$$g(m) = \frac{\int_0^{\infty} |f_m(r)|^2 r dr}{\int_0^{\infty} |f_0(r)|^2 r dr}$$

## APPENDIX B. LOG-POLAR TRANSFORM FEATURES

Psaltis and Casasent [12] have proposed log-polar transform (LPT) techniques combining a geometric coordinate transformation with the conventional Fourier transform. Smith and Devoe [13] have applied the LPT to the magnitude-squared Fourier transform of the image.

The first step in taking the LPT is to Fourier transform the input image  $f(x, y)$  to get  $F(R, \psi)$ , where  $R$  and  $\psi$  denote frequency domain radius and angle. Next, define  $\rho = \ln(R)$  and  $L(\rho, \psi) = e^{2\rho} |F(e^\rho, \psi)|^2 = R^2 |F(R, \psi)|^2$ . The Fourier transformation of the LPT is taken as

$$M(\zeta, \xi) = \int_{-\infty}^{\infty} \int_0^{\pi} L(\rho, \psi) e^{-j(\zeta\rho - \xi\psi)} d\psi d\rho$$

The  $(\zeta, \xi)$  domain is referred to here as the Mellin domain. We define the LPT features  $g_1$  and  $g_2$  as

$$g_1(m) = \frac{|M(m, 0)|}{|M(0, 0)|}, \quad g_2(n) = \frac{|M(0, n)|}{|M(0, 0)|}$$

## APPENDIX C. RING-WEDGE ENERGY FEATURES

George and his coworkers [14] developed a set of energy features which is very useful in image spectrum analysis. Wu and Stark [15] showed successful optical recognition process by using of these energy features. These features consist of energies in several ring and wedge-shaped areas in the Fourier

transform domain  $F(R, \psi)$ . Let  $\Delta_R$  and  $\Delta_\psi$  denote the step sizes for  $R$  and  $\psi$ , respectively. Define the surface area of the  $m$ th ring as  $S_r(m) = \{(R, \psi) \mid (m-1)\Delta_R \leq R \leq m\Delta_R, 0 \leq \psi \leq \pi\}$  and the surface area of the  $n$ th wedge as  $S_w(n) = \{(R, \psi) \mid 0 \leq R \leq \infty, (n-1)\Delta_\psi \leq \psi \leq n\Delta_\psi\}$ . The energy in the  $m$ th ring and the energy in the  $n$ th wedge are respectively

$$E_r(m) = \iint_{S_r(m)} |F(R, \psi)|^2 R dR d\psi, \quad E_w(n) = \iint_{S_w(n)} |F(R, \psi)|^2 R dR d\psi$$

To introduce rotation invariance, we circularly rotate  $E_w(n)$  such that

$$E_w(1) = \max_n E_w(n)$$

Since the rotation angle  $\phi$  is not always a multiple of  $\Delta_\psi$ , we normalize the ring-wedge features as

$$g_r(m) = \frac{E_r(m)}{\sum_k E_r(k)}, \quad g_w(n) = \frac{E_w(n)}{\sum_k E_w(k)}$$