Handwritten zip code recognition using an optical radial basis function classifier

M. A. Neifeld, S. Rakshit and D. Psaltis

California Institute of Technology, Department of Electrical Engineering
Pasadena, California 91125

ABSTRACT

We describe a new optical architecture for the realization of distance-based classifiers and in particular, radial basis function classifiers. This new system uses spatial multiplexing to improve computation speed while simple parallel postprocessing electronics are used to provide on-line learning in the optical neural network. This architecture is compared with a previous optical system which realized a radial basis function classifier in the solution of the handwritten digit recognition problem.

1. INTRODUCTION

We have previously reported on an optical disk-based system for the recognition of handwritten numerals.[1,2] That system made use of the large storage capacity of the optical disk ($\approx 10^{10}$ bits) in conjunction with a simple parallel readout scheme to realize a K-nearest neighbor as well as the radial basis function (RBF) classifier. Both of these classifier implementations relied on temporal multiplexing and serial postprocessing electronics for which processing time increases linearly with template library size. In this paper we describe a new optical architecture for the realization of distance-based classifiers and in particular, RBF classifiers. This new system uses spatial multiplexing to improve computation speed while simple parallel postprocessing electronics are used to provide on-line learning in the optical neural network.

2. RADIAL BASIS FUNCTIONS

In estimation and approximation theory, the RBF approach may be derived as the optimal solution to the regularized problem for the case of a specific smoothing operator.[3,4] The RBF solution defines an approximating function $\hat{f}(\underline{x})$ as a weighted sum of radially symmetric basis functions in $\Re^N$. Given a training set $T = \{\underline{t}^i, f(\underline{t}^i); i = 1, ..., M\}$ comprising a set of M points $\{\underline{t}^i \in \Re^N; i = 1, ..., M\}$ and the values of some unknown function at those points, the RBF approach specifies an estimator as

$$\hat{f}(\underline{x}) = \sum_{i=1}^{M} a_i exp(-|\underline{x} - \underline{t}^i|^2/\sigma_i^2), \tag{1}$$

where the "centers" or "templates" $\{\underline{t}^i\}$, the "widths" $\{\sigma_i\}$, and the weights $\{a_i\}$ are determined from the training set.

Since feedforward neural networks define a class of approximators, we might expect the RBF approach to correspond to an interesting class of feedforward networks. This is indeed the case and in fact, this observation leads to a useful parallel implementation of the RBF solution. Consider the network shown in Figure 1. If we define the units in the middle layer of this network as RBF units with the response function $g(\underline{x}) = exp(-|\underline{x} - \underline{t}|^2/\sigma^2)$ then the output of the final linear unit is the desired RBF approximation $z = \hat{f}(\underline{x})$. In the case of a *classification* network, the continuous valued output may be retained to indicate a confidence level or a threshold unit may be used at the output to produce a binary decision. For RBF networks, the learning problem translates to finding the centers, widths and weights which allow the network to reproduce the training set T, with minimum error.

3. OPTICAL DISTANCE COMPUTATION

The hardware implementation of a RBF network comprises two primary components: the subsystem to compute M parallel Euclidean distances and the basis function evaluation subsystem. Shown in Figure 2 is an optical system which can be used to realize the required parallel distance computation for the case of binary vectors. A similar

 0-8194-0578-7/91/$4.00

system can be used to compute the distances for continuous valued vectors; however, we will concentrate on the binary system for now. In Figure 2, an N-dimensional binary vector $\underline{x}$ is represented as a vertical intensity array in the input plane and each center $\underline{t}^i$ is stored in a vertical column of the $\underline{t}$ transparency shown. This system is "dual rail" since it requires $\underline{x}$, $\underline{t}$ and their complements $\overline{\underline{x}}$, $\overline{\underline{t}}$ respectively. We now show that using this representation, the distance computation may be performed entirely optically.

Given an input $\underline{x}$ and some center $\underline{t}^i$, we can write the Euclidean distance between these two vectors as

$$\begin{aligned} d^i &= |\underline{x} - \underline{t}^i|^2, \\ &= \sum_{j=1}^{N} (x_j - t_j^i)^2, \\ &= \sum_{j=1}^{N} d_j^i. \end{aligned}$$

We can further write the component-wise distances d_j^i in the binary case, as the exclusive or (XOR) of the component bits. That is

$$\overline{d_j^i} = x_j t_j^i + \overline{x_j}\,\overline{t_j^i}.$$

Writing d_j^i in this complement form makes the optical realization more clear. Returning to the system of Figure 2, light from the $\underline{x}$ spatial light modulator (SLM) is collimated in the x-direction and imaged in the y-direction so that immediately to the right of the transparency $\underline{t}$, the component-wise product is formed between the input and all of the centers. That is, we generate the array $\{x_j t_j^i; i = 1, ..., M; j = 1, ..., N\}$. Similarly, in the lower arm of the system the complement array $\{\overline{x_j}\,\overline{t_j^i}; i = 1, ..., M; j = 1, ..., N\}$ is formed and these two arrays are simultaneously imaged onto a contrast reversing SLM . This superposition combined with the contrast reversal yields the desired component distances d_j^i to the right of the contrast reversing SLM. The required contrast reversal may be accomplished using a number of 2D SLMs.

Returning once again to Figure 2, after the bitwise XORs are computed as described above, the desired array of distances is obtained by summing in the y-direction using the cylindrical lens shown. A simple 1D SLM can be used to represent the desired widths so that immediately to the right of the output plane shown we obtain the desired terms $\{|\underline{x} - \underline{t}^i|^2/\sigma_i^2; i = 1, ..., M\}$.

Although the system described above operates on 1D arrays of data, a 2D version which is better suited to operating on image data is also possible. This 2D extension is straightforward and involves the use of lenslet arrays for accessing the spatially multiplexed template images stored in the $\underline{t}$ and $\overline{\underline{t}}$ planes. The details of the 2D system as well as those of an extended 1D system capable of operating on continuous valued vectors are the subject of another paper and will not be discussed here; however, in order to indicate the expected performance limitations of these systems we consider the 2D binary version. If we assume that the inputs to our system are 100X100 pixel images then the number of centers in the RBF network will be limited by the SBP of the optical system. Realization of 900 centers will require a template mask with 3000X3000 pixels which in turn demands an optical system with SBP=$9X10^6$. This would be the largest feasible implementation. Notice that although the contrast reversal plane must have large SBP, the output plane requires only SBP equal to the number of centers. This is an attractive characteristic of the present system since on-line learning will require a programmable SLM in this plane for σ adaptation. In the present example, this SLM would be required to have only 30X30 pixels.

4. BASIS FUNCTION EVALUATION

Having defined the optical distance computer in the previous section we turn our attention to the second primary component of the optical RBF implementation. This component performs the basis function evaluation. Notice that the basis function evaluation requires only point operations in the plane of distances. Further, if we consider the case of a single output neuron (i.e., $f(\underline{x}) : \Re^N \rightarrow \Re^1$) then to complete the RBF computation after the distance computer requires *only* point operations followed by a global sum. With this observation in mind we propose the optoelectronic postprocessing chip shown in Figure 3.

The chip shown consists of an array of modules each module comprising a photodetector to detect the output of the optical distance computer, analog multipliers to realize the required width and output weighting and an exponentiation unit to realize the basis function evaluation. The output of each such module is summed on a common line to generate the network response. We should note here that all the required functions in a module are easily achieved using analog VLSI or if more precision is required, the photodetector may be followed by an A/D converter and each module could then be implemented in digital electronics. The 2D extension of this postprocessing chip is once again straightforward and since there are no intermodule communication requirements, connectivity issues in the 2D arrangement do not arise. All computation in this postprocessing chip is local excepting the final sum. Also note that this implementation is quite flexible and allows for the realization of a variety of different basis functions as well as supporting a useful on-line learning algorithm which will be discussed further in the next section.

The above "all electronic" postprocessing chip is particularly well suited to the case of a network with only one output. For the case of multiple outputs we propose two alternative systems. If the number of outputs is relatively small (≈ 10) then the most attractive alternative is simply the use of multiple postprocessing chips. This approach retains the simplicity and flexibility of the VLSI implementation. Alternatively, if the number of outputs is large, we may consider a hybrid approach wherein each $[e^{-x}]$ box in Figure 3 is followed by a modulating pad allowing the exponentially weighted distances to be read out optically using liquid crystal modulators for example. In this way an efficient, optical implementation of the output layer is facilitated. This approach has the advantage of providing scalability in terms of output units while retaining much of the convenience of the VLSI implementation. In this system, update of the output weights during a learning cycle becomes somewhat more difficult; however, the use of photorefractive holograms in the realization of the output layer will make such a learning scheme possible.

5. LEARNING

The effective implementation of a learning algorithm is a common stumbling block in both electronic and optical neural network architectures. In this section we suggest a learning algorithm for RBF networks which is suitable for implementation using the optoelectronic hardware we have described. Associated with each of the postprocessing stages (i.e., all electronic and hybrid) is an implementation of the on-line learning algorithm. We will describe the all electronic single output implementation here.

Referring to Equation 1 for the RBF network response function, we can define a criterion function for the "goodness" of an RBF network as

$$E = \sum_{i=1}^{M} (\hat{f}(\underline{x}^i) - f(\underline{x}^i))^2,$$

$$\sum_{i=1}^{M} E_i^2,$$

where E_i is just the error between the actual and desired network responses in the presence of training vector $\underline{x}^i$. E is just the conventional "sum of squared error" function evaluated over the entire training set. If we assume that the training set T is fixed and that each training vector will be used as a single RBF center as before, then the learning procedure reduces to finding $\{\sigma_i\}$ and $\{a_i\}$ to minimize the error E. A simple gradient decent procedure is a candidate algorithm for the minimization of E. Using this procedure we can write expressions for the update of the network parameters σ_p and a_p in response to the error measured for a single input training vector $\underline{x}^l$. These are

$$\Delta(1/\sigma_p)^2 = -\alpha_\sigma E_l |\underline{x}^l - \underline{t}^p|^2 a_p exp(-|\underline{x}^l - \underline{t}^p|^2/\sigma_p^2), \quad (2)$$

$$\Delta(a_p) = \alpha_a E_l exp(-|\underline{x}^l - \underline{t}^p|^2/\sigma_p^2), \quad (3)$$

where α_σ and α_a are acceleration constants for the width and output weight updates respectively. Notice that these expressions define a "backward error propagation" type of rule for RBF networks. In this learning algorithm however, no special backward response function is required for the RBF units owing to the fact that the $exp(x)$ function is its own derivative. All signals required to compute the updates defined in Equations 2 and 3 above are present in the *forward* path of the network. Furthermore we observe that all required learning signals are present in the electronic portion of the proposed implementation and that no intermodule communication is necessary. The implication of

these observations is that rapid, parallel update of all network parameters can be realized with a simple modification of the postprocessing module presented earlier. In Figure 4 we show a block diagram of the modified postprocessing module. By incorporating the additional "local" connections shown and adding accumulation registers to the $[(1/\sigma)^2]$ and $[Xa]$ blocks, the on-line parallel RBF learning algorithm can be realized with negligible increase in overall circuit complexity over the non-adaptive system.

6. CONCLUSIONS

In a previous paper we reported on a serial version of the optical RBF classifier which makes use of optical disk technology to realize a network containing 650 centers and 10 output neurons.[1] This system was trained off-line, with the handwritten numerals 0-9 and achieved a recognition rate in simulation of 89% on a 300 element testing set. Similar performance (91% recognition rate) was achieved using the same off-line training procedure in an RBF network with 2000 center, trained on segmented zip code data obtained from the U.S. post office database. The optical disk based 650 center system achieved a recognition rate of 83%. In that work it was found that factors such as nonuniform disk reflectivity, nonuniform illumination and finite contrast were all significant contributors to a 28% RMS error in the optical distance computation. Furthermore, since this large distance error resulted in only a 6% degradation in recognition performance, the RBF approach was seen to be robust in the presence of such errors. We might expect an on-line learning scheme in which optical system imperfections are present during the learning phase, to provide compensation for those imperfections and result in a recognition rate closer to the simulation value.

In this paper we have described several alternative approaches to an RBF optical hardware implementation. The systems described here are well suited to the realization of an on-line learning procedure based on the gradient decent algorithm given herein. Through the combination of parallel optical distance computations and electronic postprocessing components, it is possible to design optical neural systems whose on-line learning capabilities might improve their robustness to optical system imperfections.

7. ACKNOWLEDGEMENTS

This work is supported by the U.S. Army Research Office and the Defense Advanced Research Projects Agency.

8. REFERENCES

[1] M. A. Neifeld, S. Rakshit and D. Psaltis, "Optical Disk Implementation of Radial Basis Classifiers," *Proceedings of SPIE 1990 International Symposium on Optical and Optoelectronic Applied Science and Engineering*, SPIE paper 1347-02, San Diego, CA., 1990.

[2] D. Psaltis, M. A. Neifeld, A. A. Yamamura and S. Kobayashi, "Optical Memory Disks in Optical Information Processing Systems," *Applied Optics*, Vol. 29, No. 14, May, 1990.

[3] T. Poggio and F. Girosi, "Networks for Approximation and Learning," *Proceedings of IEEE*, Vol. 28, No. 9, September, 1990.

[4] J. Moody and C. Darken, "Fast Learning in Networks of Locally Tuned Processing Units," *Neural Computation*, Vol. 1, pp. 281-294, 1989.

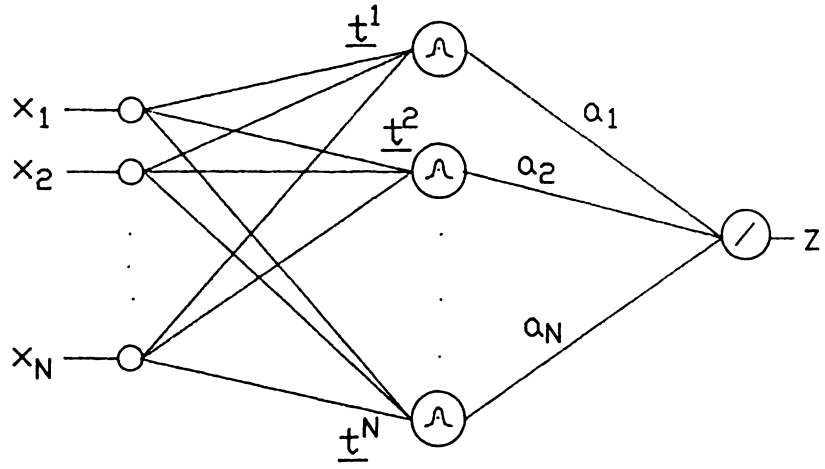

Figure 1 : Radial Basis Function Network

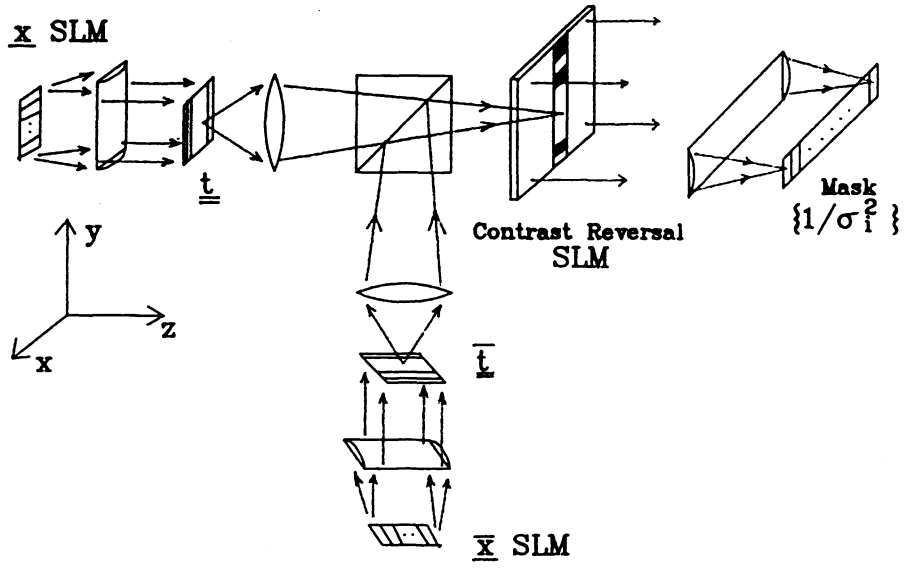

Figure 2 : Optical Distance Computer

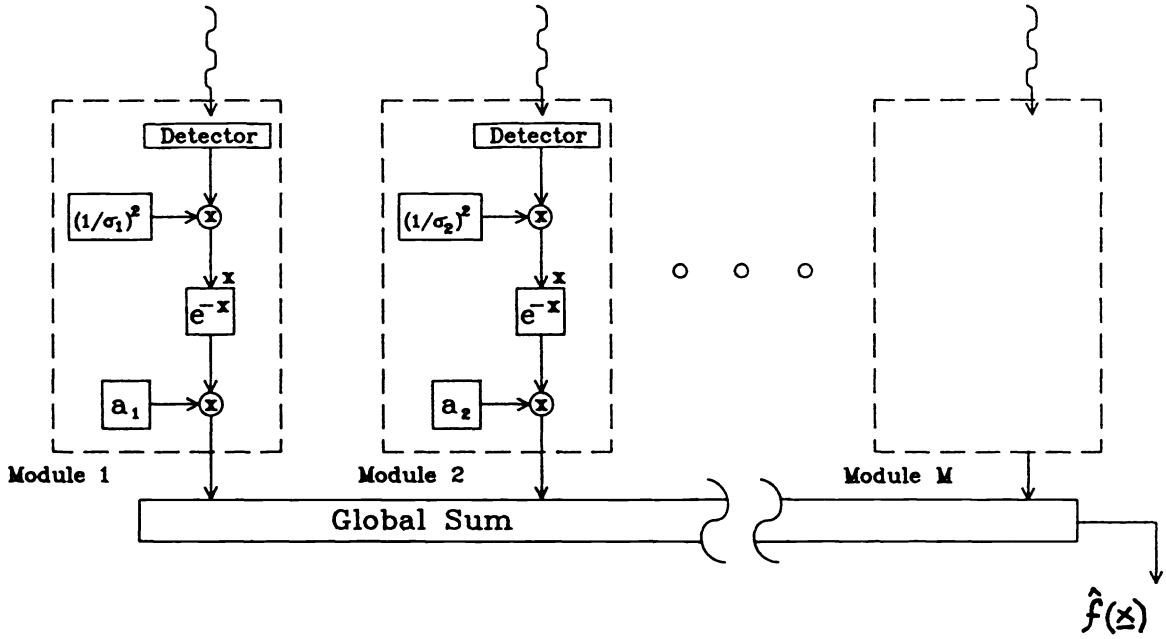

Figure 3 : Optoelectronic Postprocessing Chip

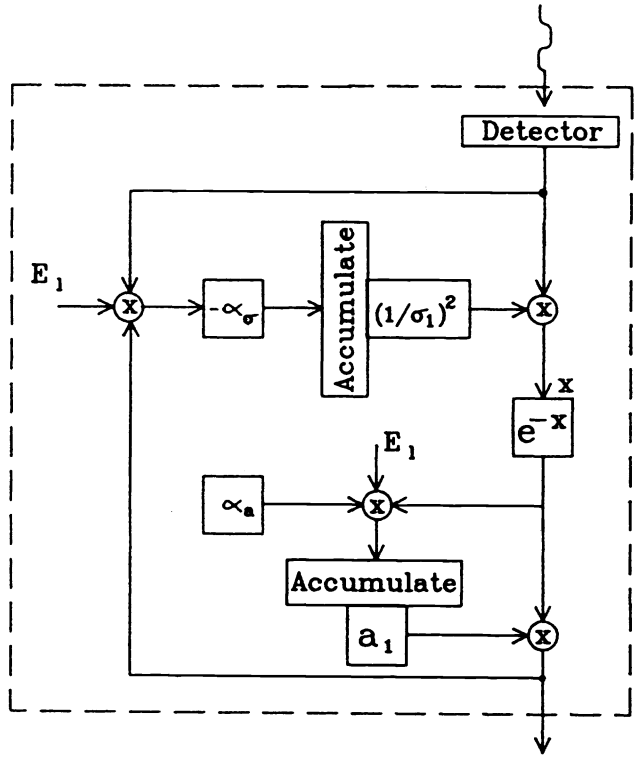

Figure 4 : On-Line Learning Postprocessing Module