A Class of Instantaneously Trained Neural Networks

Subhash Kak[*]
Department of Electrical & Computer Engineering,
Louisiana State University,
Baton Rouge, LA 70803-5901

May 7, 2002

Abstract

This paper presents FC networks that are instantaneously trained neural networks that allow rapid learning of non-binary data. These networks, which generalize the earlier CC networks, have been compared against Backpropagation (BP) and Radial Basis Function (RBF) networks and are seen to have excellent performance for prediction of time-series and pattern recognition. The networks can generalize using soft or hard decisions.

1 Introduction

The popular Backpropagation (BP) and Radial Basis Function (RBF) neural networks require iterative training. BP networks sometimes don't converge or take too long to train to be useful in real-time applications. BP networks have been proposed as models of biological memory, but given the time it takes these networks to learn, it is clear they could never learn short-term, instantaneously-learned, memory.

[*]E-mail address: kak@ece.lsu.edu

Short-term and long-term memories form a complementary pair in biological functioning. Similarly, quick learning using artificial neural networks should be useful for many engineering applications. It was the motivation to model short-term memory that led to the development of the corner classification (CC) family of networks [1,2] that learn instantaneously and have very good generalization performance. Further work was motivated by an invitation from Karl Pribram to speak on different languages of the brain which led me to analyze the associative apects of memory at some length [3]. Further development of these ideas followed in due course [4,5,7].

These networks learn example patterns and then generalize points that lie in a sphere of fixed range around them, without a specific strategy for other points that lie outside the sphere. It has been shown that these networks are hardware friendly and suited for implementation in reconfigurable computing using fine grained parallelism [9]. Although the CC networks have found many applications in engineering and business [5], they suffer from the disadvantage that their input and output must be discrete.

Another disadvantage of the CC networks is that the input is best presented in a unary code, which increases the number of input neurons considerably. Furthermore, the degree of generalization achieved at each trained node is, in the non-adaptive version of the CC network, kept constant. In reality, the data might require that the amount of generalization vary from node to node. These characteristics somewhat limit the applicability of CC networks.

In the present paper we describe a generalization of the CC network that we call the FC network [6]. This network can operate on real data directly and offer great flexibility as compared to the CC networks. The FC networks are described in detail elsewhere [6,8]. Here our objective is to focus on their fundamental structure to highlight how they are distinct from other approaches to neural network design and learning.

2 The Network

The FC network maps data to the nearest neighbor within whose sphere of generalization the data falls, and if that is not the case it performs an interpolation based on k nearest neighbors. As k becomes large, the FC generalization can be shown to approach the Bayes performance [6]. It uses the fully-connected feedforward network architecture consisting of a layer of input nodes, a layer of hidden neurons, a rule base, followed by an output layer (Figure 1). The number of output neurons is determined by the problem specifications. A network with multiple output neurons can always be separated into multiple single-output networks.

An instantaneously trained neural network must, by definition, convert the incoming pattern into corresponding weights without intensive computations. This rules out any technique that is based on reduced-dimension representation, and the network must depend on a linear mapping of the data. It is this linearity requirement that compels the use of unary mapping in CC networks. However, the transformation of the data within the network cannot be entirely linear because that would make it impossible for the networks to generalize. The networks have a nonlinear generalizing region and elsewhere they perform nonlinear interpolation mapping.

Making a transition from binary to non-binary networks, we find that abandoning the inefficient unary mapping of data we are able to use real values as they stand. Surprisingly, this generalization allows us to cut down on the size of the network. The price that is paid is in terms of the extra computations necessary to separate patterns in the signal space.

We want the network to work in two steps:

1. A basic architecture based on the exemplar patterns

2. A tuning of the network to make the patterns consistent

The CC networks merely learn the exemplar patterns without a systematic procedure for tuning of the generalization process.

In the FC networks, input data are normalized to the range $[0, 1]$ and presented to the network in the form of an R-element long continuous-valued vector $x = (x_1, x_2, ..., x_R)$, where R is determined by the problem specification. Like the CC network, the number of hidden neurons S in an FC network is equal to the number of training samples the network is required to learn.

Each hidden neuron i $(i = 1, 2, ..., S)$ is associated with a weight vector $w_i = (w_{i,1}, w_{i,2}, ..., w_{i,R})$, where $w_{i,j}$ is the weight from source x_j in the vector to the ith hidden neuron.

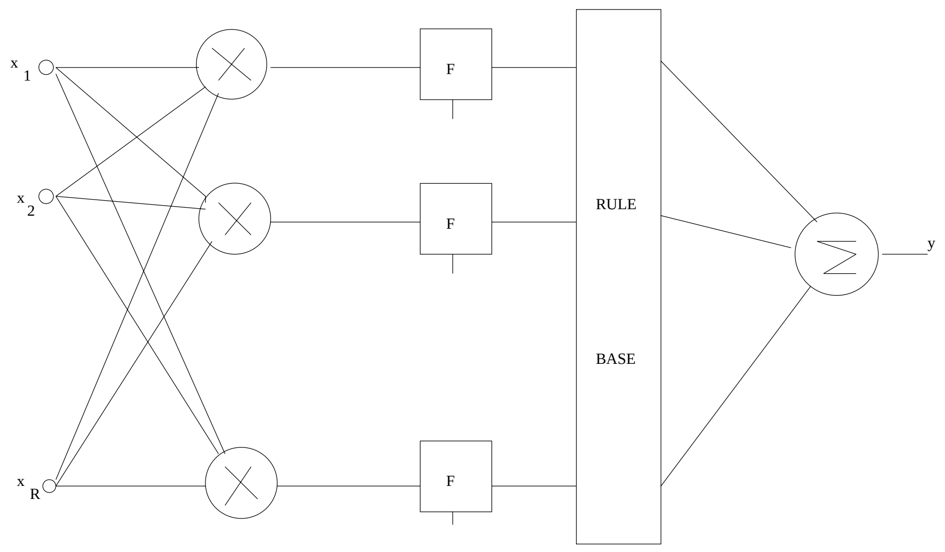

Figure 1: An FC Network. The square boxes, F, adjust the generalization radii

Each hidden neuron i first computes the normalized Euclidean distance d_i between its weight vector w_i and the test vector x. This distance together with r_i, the radius of generalization for this hidden neuron, constitute the inputs to the activation function F which determines the hidden neuron output h_i. The output of all hidden neurons taken together forms the distance vector $h = (h_1, h_2, ...h_S)$ that gives a measure of similarity between the test vector and each of the training vectors the network has already learned.

The rule base enables the FC network to generalize. It consists of a set of IF-THEN rules that operates on the distance vector h to produce a membership grade vector $\mu = (\mu_1, \mu_2..., \mu_S)$ that indicates to what degree the test vector x belongs to each of the network output classes. The output neuron then computes the dot product between output weight vector $u = (u_1, u_2, ..., u_S)$ and the membership vector μ to produce the generalized network output y corresponding to test vector x. Training of an FC network involves two distinct steps. The first step determines the input and output weights while the second step finds the radius of generalization for each hidden neuron.

Unlike the neurons in a CC4 network, the input vector x, the weight vector w, as well as the scalar output h in the FC neuron are all continuous-valued quantities. Quantity d is the Euclidean distance between x and w. The parameter r is called the radius of generalization, a term borrowed from the CC4 network. Its purpose is to allocate to this hidden neuron, an area of generalization in the input space with radius r and its center at w. Unlike the CC4 network where the same r applies to the whole network, the radius of generalization in an FC network is individually determined for each hidden neuron during training.

The network can be trained with just two passes of the training samples. The first pass assigns the synaptic weights for the input and output layers. The second pass determines the radius of generalization r for each training sample. The notion of radius of generalization for each training vector provides a basis for the network to switch between a 1NN classifier and a kNN classifier during generalization. The network behaves like a 1NN classifier when the input vector falls within the area of generalization of a training vector, and a kNN classifier otherwise. This enables the network to benefit from the strength of both classifiers.

The network can generalize in a variety of ways. In earlier work [6,8], we have considered soft generalization which makes it possible to interpolate outside of the generalization spheres. This is shown in Fig 2(a) for three learned examples labeled 1, 2, and 3. The radius of these spheres is exactly one-half of the distance to the nearest other exemplar pattern. But the entire space can be divided into specific classes, using hard decision boundaries, as in Figure 2(b). As to which of the two methods to use would depend on the nature of the problem and the geometry of the output values.

Example Consider the following input-output training samples:

Sample	Input				Output
1	2	3	4	-1	7
2	1	-1	2	-3	4
3	3	0	1	8	9

The Euclidean distances between the samples are: $d_{12} = 5,\ d_{13} = 10,\ d_{23} = 11.27$. The radius of generalization for a node is one-half the least distance to the other samples. Therefore, the radii to be associated with the three hidden nodes are $2.5,\ 2.5,\ 5$, respectively.

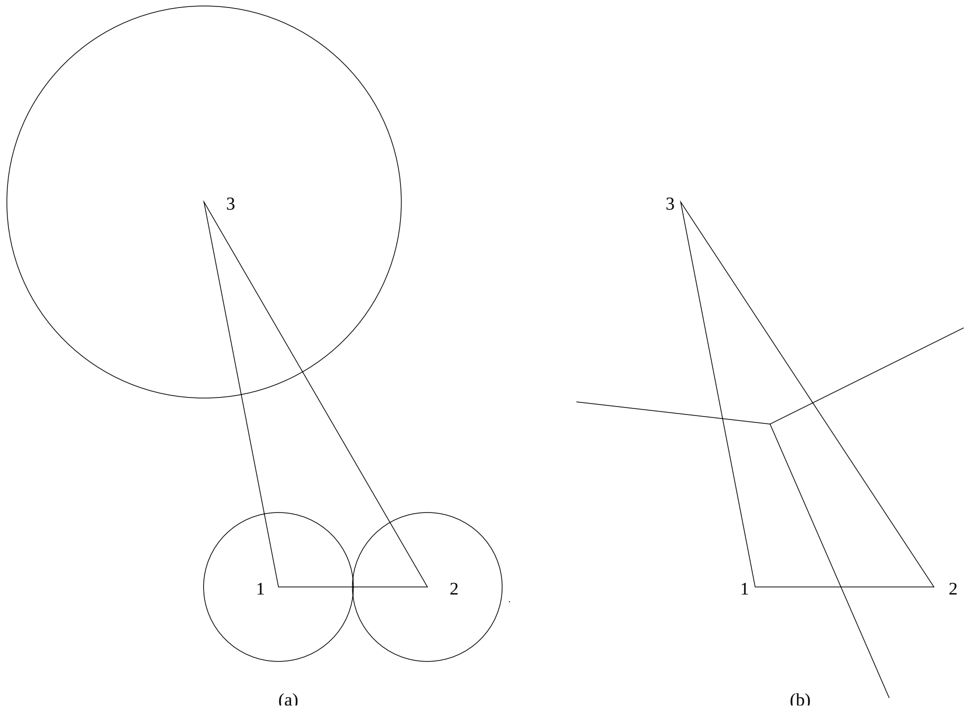

Figure 2: (a) Soft generalization together with interpolation, (b) hard generalization with separated decision regions

Now, consider an input vector $X = (1, 2, 3, 4)$, whose output we wish to compute. This vector is at the Euclidean distance of $d_1 = 5.29$, $d_2 = 7.68$, and $d_3 = 5.29$ from the three stored vectors. Since these distances are larger than the radii of generalization of the three vectors, we compute the interpolation weights $\mu_i = \frac{1/d_i}{1/d_1+1/d_2+1/d_3}$

This gives us $\mu_1 = 0.372$, $\mu_2 = 0.256$, $\mu_3 = 0.372$. So, the output is: $0.372 \times 7 + 0.256 \times 4 + 0.372 \times 9 = 6.976$.

This is an example of the use of soft generalization.

The FC network possesses generalization characteristics that compare favorably with other neural networks such as the BP and RBF networks [6,8]. The design of an FC network for any real-world problem is very easy compared to BP and RBF networks. Therefore, it offers an attractive alternative to the other networks.

3 Concluding Remarks

The paper has presented the outline of a method for instantaneous learning of patterns by a neural network that works both for binary and non-binary data. This method is a direct generalization of the CC family of networks where the actual distances between the exemplars are computing to determine the radius of generalization to be associated with each hidden node.

The generalization strategy must depend on the nature of the data. Further research on soft generalization options related to problem geometry needs to be carried out since the earlier work [6,8] has only considered a specific method.

The method described in this paper has the interesting property that the tuning of the generalization procedure is carried out later. The tuning could include reduction in the dimensions of the problem space, which are not described in this paper. It can also be enhanced by means of node pruning techniques. Perhaps, it is tuning of this kind that transforms a short-term memory into a long-term one.

Acknowledgement

This paper is a summary of the plenary lecture delivered at the JCIS 2002 in Durham, North Carolina, in March 2002.

References

[1] S. Kak, On training feedforward neural networks. *Pramana -J. of Physics* 40 (1993) 35-42.

[2] S. Kak, New algorithms for training feedforward neural networks. *Pattern Recognition Letters* 15 (1994) 295-298.

[3] S. Kak, The three languages of the brain: quantum, reorganizational, and associative. In *Learning as Self-Organization*, K. Pribram and J. King (eds.). Lawrence Erlbaum, Mahwah, 1996, pp. 185-219.

[4] S. Kak, "On generalization by neural networks," *Information Sciences* 111 (1998) 293-302.

[5] S. Kak, "Better web searches and prediction with instantaneously trained neural networks," *IEEE Intelligent Systems* 14(6) (1999) 78-81.

[6] K.W. Tang, *Instantaneous Learning Neural Networks.* Ph.D. Dissertation, LSU, 1999.

[7] K.W. Tang and S. Kak, "A new corner classification approach to neural network training," *Circuits, Systems Signal Processing* 17 (1998) 459-469.

[8] K.W. Tang and S. Kak, "Fast classification networks for signal processing," *Circuits, Systems Signal Processing* 21 (2002) 207-224.

[9] J. Zhu and G. Milne, "Implementing Kak neural networks on a reconfigurable computing platform," In FPL 2000, LNCS 1896, R.W. Hartenstein and H. Gruenbacher (eds.), Springer-Verlag, 2000, pp. 260-269.