Communicated by David Lowe

Improving the Generalization Properties of Radial Basis Function Neural Networks

Chris Bishop
Neural Networks Group, AEA Technology,
Harwell Laboratory, Oxfordshire OX11 0RA, United Kingdom

An important feature of radial basis function neural networks is the existence of a fast, linear learning algorithm in a network capable of representing complex nonlinear mappings. Satisfactory generalization in these networks requires that the network mapping be sufficiently smooth. We show that a modification to the error functional allows smoothing to be introduced explicitly without significantly affecting the speed of training. A simple example is used to demonstrate the resulting improvement in the generalization properties of the network.

1 Introduction

Radial basis function (RBF) neural networks (Broomhead and Lowe 1988) provide a powerful technique for generating multivariate, nonlinear mappings. Unlike the widely used technique of error backpropagation (Rumelhart and McClelland 1986) the learning algorithm for RBF networks corresponds to the solution of a linear problem. The training of the network is therefore a fast procedure.

An important consideration in setting up an RBF network is the choice of the number and centers of the radial basis functions (i.e., the hidden units). The most natural choice is to let each data point in the training set correspond to a basis function center. In this case the number of degrees of freedom in the network equals the number of items of data, and the network function fits exactly through each data point. If the data have a regular behavior, but are contaminated by noise, the network will learn all the details of the individual data points, rather than representing the underlying trends in the data. This phenomenon is sometimes called overfitting. The resulting network function often has poor generalization properties as a result of the rapid oscillations that usually characterize an overfitted function. One procedure for damping out these oscillations, referred to as curvature-driven smoothing, has been developed earlier in the context of networks trained by error backpropagation (Bishop 1990). Here we show that an analogous technique can be applied in the case of RBF networks, and that the resulting trained networks do indeed exhibit improved generalization.

Neural Computation **3, 579–588** (1991)

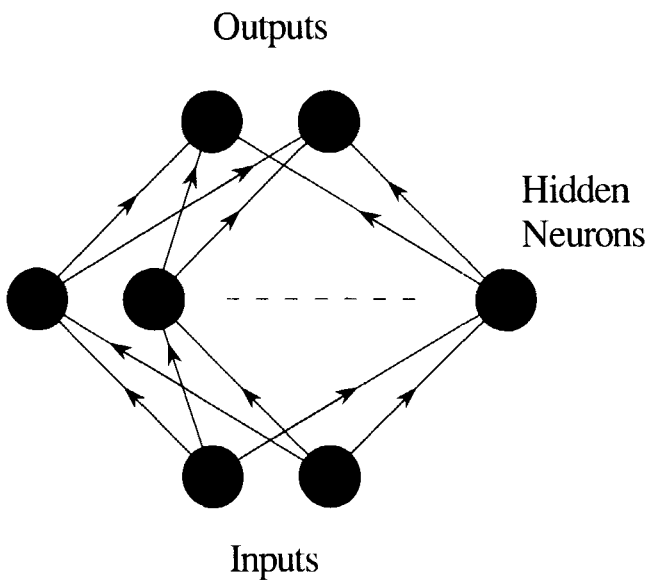

Figure 1: Architecture of a radial basis function network.

An introduction to RBF networks is given in Section 2. In Section 3 the technique of curvature-driven smoothing is developed in the context of RBF networks, and results from the application to a simple problem are presented in Section 4. A brief summary is given in Section 5.

2 Radial Basis Function Networks

Here we review briefly the central features of radial basis function networks. For a more extensive discussion see Broomhead and Lowe (1988). The network has a three layer feedforward architecture as shown in Figure 1. Input vectors $\mathbf{x}$ are propagated to the hidden units (hidden neurons) each of which computes a hyperspherical function of $\mathbf{x}$, so that the output of the ith hidden unit is given by

$$\phi_i = \phi(\| \mathbf{x} - \mathbf{y}_i \|) \tag{2.1}$$

where $\mathbf{y}_i$ is the center of the radial basis function for unit i, and $\| \ldots \|$ denotes a distance measure that is generally taken to be the Euclidean norm. The nonlinear function ϕ can be chosen in a variety of ways and

can in principle vary from one hidden unit to the next. For the examples shown later we have taken a gaussian nonlinearity:

$$\phi(x) \equiv \exp\{-x^2/\sigma^2\} \tag{2.2}$$

The outputs of the network are formed from the weighted sum of the outputs from the hidden units:

$$z_i = \sum_j w_{ij}\phi_j + \theta_i \tag{2.3}$$

where the synaptic weights w_{ij} and the biases θ_i are adaptive variables that are set during the learning phase. The bias terms can be absorbed into the weight matrix by introducing an extra hidden unit whose output $\phi_k = 1$. Training data are supplied to the network in the form of pairs $\mathbf{x}_p$, $\mathbf{t}_p$ of input and target vectors, where $p = 1, \ldots, P$ labels the individual training pairs. The learning algorithm aims to minimize the sum-of-squares error defined by

$$E^S = \frac{1}{2}\sum_p \sum_i (z_{ip} - t_{ip})^2 \tag{2.4}$$

where $z_{ip} = z_i(\mathbf{x}_p)$ denotes the output of unit i when the network is presented with input vector $\mathbf{x}_p$. At a minimum of E^S we have

$$\frac{\partial E^S}{\partial w_{ij}} = 0 \tag{2.5}$$

Together with equation 2.3 (and omitting the explicit bias terms) this gives

$$\sum_p \left\{ \sum_k w_{ik}\phi_{kp} - t_{ip} \right\} \phi_{jp} = 0 \tag{2.6}$$

where $\phi_{jp} = \phi_j(\mathbf{x}_p)$. This can be written in the form

$$\sum_k w_{ik} M_{kj} = \sum_p t_{ip}\phi_{jp} \tag{2.7}$$

where the square matrix $\mathbf{M}$ is defined by

$$M_{kj} \equiv \sum_p \phi_{kp}\phi_{jp} \tag{2.8}$$

Note that $\mathbf{M}$ is the covariance matrix of the transformed data (for data with zero mean). Provided $\mathbf{M}$ is not singular, we can compute $\mathbf{M}^{-1}$ (in practice using singular value decomposition), and hence solve equation 2.7 to give

$$w_{ij} = \sum_k (\mathbf{M}^{-1})_{kj} \left\{ \sum_p \phi_{kp} t_{ip} \right\} \tag{2.9}$$

Michelli (1986) has shown that for a large class of functions ϕ the matrix M is nonsingular provided the data points are all distinct. For nonsingular M, the quantity $(\phi^T\phi)^{-1}\phi^T$, which appears implicitly in equation 2.9, is the Moore–Penrose pseudoinverse of the matrix ϕ (Golub and Kahan 1965). In the case where the number of basis functions equals the number of training data points, the matrix ϕ is square, and the pseudoinverse of ϕ reduces to the usual inverse. The minimum of E^S then occurs at $E^S = 0$, and the function generated by the trained network passes exactly through every data point.

One of the great advantages of RBF networks is that the learning algorithm involves the solution of a linear problem, and is therefore fast. Due to the nonlinearity of the basis functions, however, the network can generate complex nonlinear mappings. In principle learning strategies could be devised that involve changes also in the location and form of the radial basis functions. The advantages of a linear learning algorithm would then be lost, however.

The centers $\mathbf{y}_i$ of the basis functions can be chosen in a variety of ways. A natural choice would be to take the $\mathbf{y}_i$ to be the input vectors $\mathbf{x}_p$ from the training data set, or a subset of these in the case where the number of hidden units is less than the number of training data points. If the network is to be used as a pattern classifier the number of basis functions is generally taken to be significantly larger than the number of input units. The hidden units then nonlinearly map input vectors into a space of higher dimension. The problem may be linearly separable in this higher space even when it is not linearly separable in the original space. In this case the single layer of modifiable weights between hidden and output units is sufficient to give correct classification. In this paper we are interested primarily in continuous mappings between input and output variables.

3 Curvature-Driven Smoothing in RBF Networks

The situation in which the network mapping passes exactly through each training data point is generally not desirable, even though this gives $E^S = 0$. In many practical applications of neural networks the available set of training data will be noisy. If the network mapping fits the data exactly, the capability of the network to generalize, that is to produce an acceptable output when a novel input is applied, will often be poor. This arises from the rapid oscillations in the network function that generally are needed for the function to fit the noisy data. The situation is analogous to the problem of overfitting which can occur when curve fitting using high order polynominals.

To improve the generalization capability of the network it is necessary for the network mapping to represent the underlying trends in the data, rather than fitting all of the fine details of the data set. One way

in which this can be achieved is to reduce the number of degrees of freedom by using fewer hidden units. Although this leads to a smaller network, it is not clear how the basis function centers should be chosen. One possibility is to take a subset of the input vectors from the training data. The subset may be chosen randomly, or by a more systematic elimination procedure starting with a full-sized network (Admoaitis *et al.* 1990). Another procedure for choosing the basis function centers is to use a self-organizing neural algorithm such as the "topology preserving feature map" (Kohonen 1988). If the quantity of training data available is at all limited, however, it may be undesirable to eliminate potential basis function centers, particularly if there are regions of the input space where the data are relatively sparse.

We consider here an alternative procedure for avoiding the overfitting problem in RBF networks. The full set of radial basis functions, whose centers correspond to the input vectors from the training data, is retained. An additional term is added to the error measure whose role is to damp out the rapid oscillations in the network function that are characteristic of overfitting, while retaining the longer wavelength variations describing the underlying nonlinear trends in the data. The total error function then becomes

$$E = E^S + \lambda E^C \tag{3.1}$$

where E^S is the standard sum-of-squares error given by equation 2.4 and E^C is arranged to be large for functions with rapid oscillations. The parameter λ in equation 3.1 controls the degree to which the network function is smoothed. This approach, known as regularization, is commonly used in a number of other fields for tackling "ill-posed" problems (Tikhonov and Arsenin, 1977). Poggio and Girosi (1990), starting with the concept of regularization, have derived an approximation scheme that includes radial basis function networks as a special case, thus demonstrating a close relation between these two techniques. Regularization terms also arise when considering the effects of noise on the input data in least squares functional approximation, as discussed in Webb (1991). A regularization technique, referred to as curvature-driven smoothing, has also been applied to neural networks trained by error backpropagation (Bishop, 1990).

The functional E^C in equation 3.1 will be chosen to have the following form:

$$E^C = \frac{1}{2}\sum_p \sum_i \left\{ \sum_n \left(\frac{\partial^2 z_{ip}}{\partial x_n^2} \right)^2 \right\} \tag{3.2}$$

where n labels the input unit. This choice for E^C has the required property of penalizing functions with large second derivatives and, most importantly, is bilinear in the synaptic weights. Thus the great advantage of RBF networks, namely the linear learning algorithm and consequent speed of training, will be retained.

If we now minimize E with respect to the weights $\{w_{ij}\}$ we obtain

$$\sum_p \left\{ \left(\sum_k w_{ik}\phi_{kp} - t_{ip} \right) \phi_{jp} + \lambda \left(\sum_n \left[\sum_k w_{ik} \frac{\partial^2 \phi_{kp}}{\partial x_n^2} \right] \frac{\partial^2 \phi_{jp}}{\partial x_n^2} \right) \right\} = 0 \qquad (3.3)$$

Rearranging terms gives

$$\sum_k w_{ik}\widetilde{M}_{kj} = \sum_p t_{ip}\phi_{jp} \qquad (3.4)$$

which is analogous to equation 2.7, with $\widetilde{M}$ defined by

$$\widetilde{M}_{kj} \equiv \sum_p \left\{ \phi_{kp}\phi_{jp} + \lambda \sum_n \left(\frac{\partial^2 \phi_{kp}}{\partial x_n^2} \frac{\partial^2 \phi_{jp}}{\partial x_n^2} \right) \right\} \qquad (3.5)$$

Equation 3.4 can now be solved using the same techniques as for equation 2.7.

The appropriate value for λ will be problem dependent. It should not be chosen too large since this will smooth the network function too much and lead to a deterioration in the ability of the network to generalize. Results presented in the next section suggest, however, that the performance of the network may be fairly insensitive to the precise value of λ.

The form of E^C given by equation 3.2 treats each input–output unit pair on an equal footing. It thus presupposes that the input (and output) variables have been rescaled to span a similar range of values. As an alternative, suitable scaling factors c_{in} for each input–output unit pair can be included in equation 3.2.

4 Simulation Results

We now illustrate the ideas introduced in the previous section with a simple example. Consider a network with a single input unit and a single output unit. Data are generated from the function

$$z = 0.8 \sin(2\pi x) \qquad (4.1)$$

sampled at 25 equally spaced values of x in the range $(0, 1)$, and perturbed with $\pm$ 20% random noise. A similar set of test data was generated by sampling equation 4.1 at intermediate values of x, and again perturbing with $\pm$ 20% noise. The number of basis functions is chosen to equal the number of training data points. Gaussian basis functions of the form of equation 2.2 are used, and the basis function centers are taken to coincide with the training data input vectors.

Figure 2 shows the training data together with the network function that results from training the network without any smoothing. The function fits each data point exactly, and the rapid oscillations (with corresponding high curvature) that are characteristic of overfitting are clearly seen.

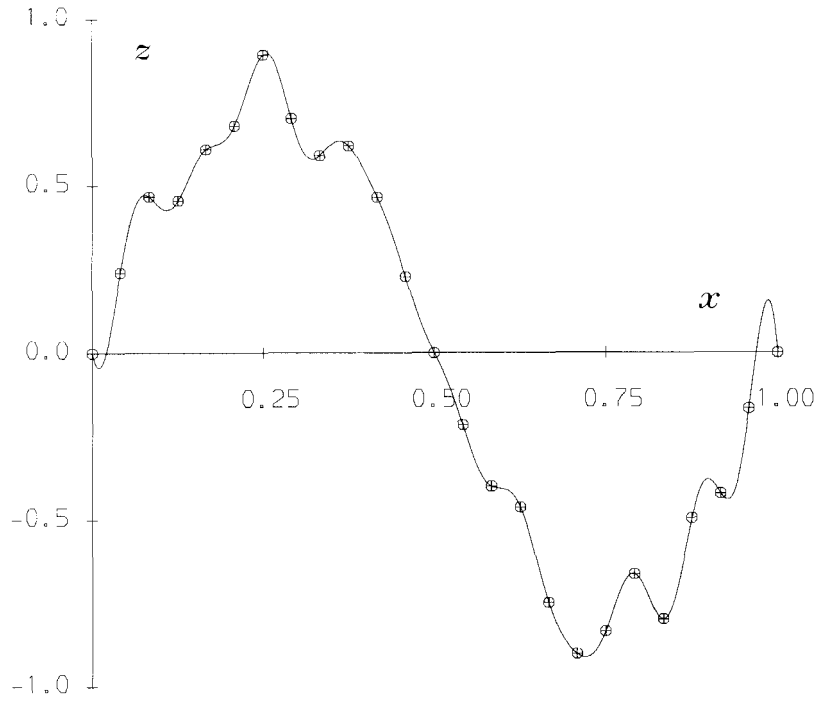

Figure 2: Training data generated from the function $z = 0.8\sin(2\pi x)$ and perturbed with $\pm$ 20% noise, together with a plot of the network function obtained without smoothing.

The effect of introducing a smoothing term, with a small value of λ, is to increase the error with respect to the training data, while reducing the test data error. This is illustrated in Figure 3 in which the mean square error E^S is plotted as a function of $\ell n\lambda$ for both training and test data. The fall in the test data error indicates that the network is better able to generalize. Larger values of λ result in oversmoothing, and the error with respect to the test data increases again.

For this example the optimum value of λ, corresponding to the minimum of the test data error, is given by $\lambda = 8.3 \times 10^{-6}$. The corresponding network function is plotted in Figure 4. At this value of λ the short scale oscillations are completely suppressed. The minimum value of the test error is close to the value 0.004 obtained by comparing the test data with the original function in equation 4.1, showing that the network has good generalization properties when λ is set to the optimum value.

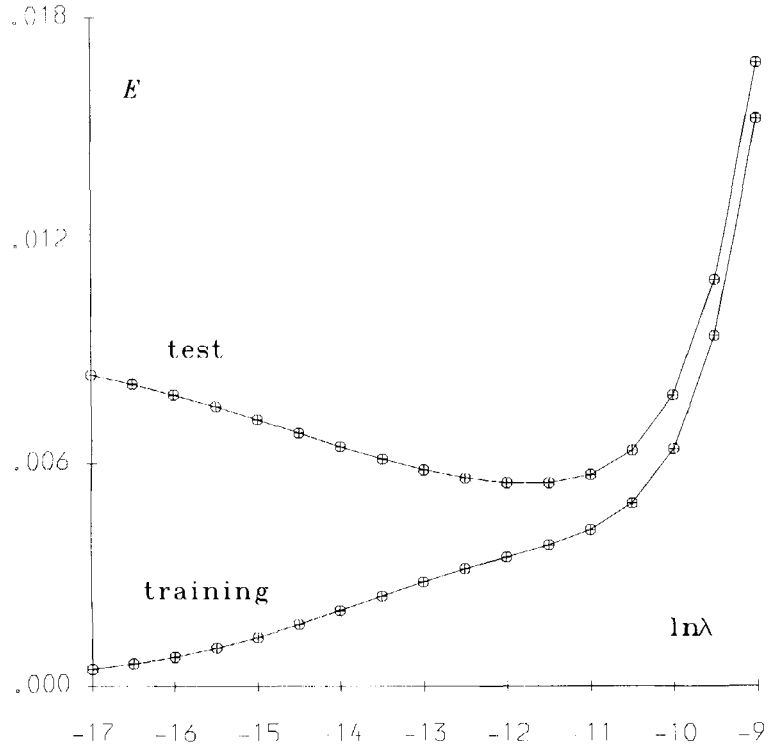

Figure 3: Mean square error for training data (lower curve) and test data (upper curve) versus $\ln \lambda$.

Although the appropriate value for λ must be determined by experiment, Figure 3 indicates that variations in λ of about an order of magnitude (note the logarithmic scale of the abscissa) have little effect on the test data error for this problem. The parameter σ in equation 2.2, which governs the width of the gaussian functions, also has to be chosen appropriately. Too small a value leads to a hidden unit response which is highly localized, making it difficult to generate smooth network functions. At too large a value, the matrix M becomes ill-conditioned. A suitable choice would allow the gaussians to span a number of data points, and a value of $\sigma = 0.1$ was used in the above example.

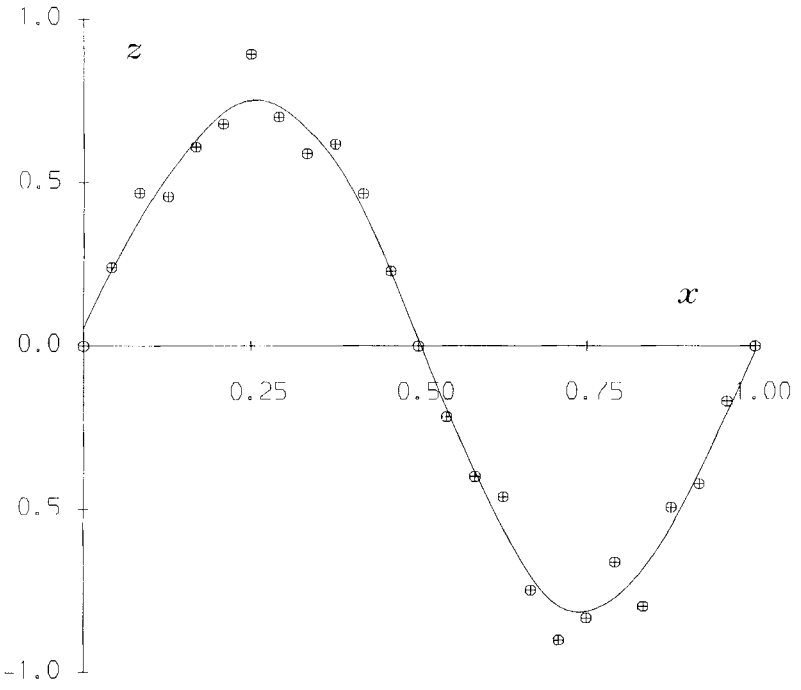

Figure 4: Plot of the network function obtained with a smoothing term and with $\lambda = 8.3 \times 10^{-6}$.

5 Summary

In this paper we have described a practical procedure for improving the generalization properties of radial basis function neural networks. The performance of the network for new data (i.e., data not used during training) can be controlled by varying a single parameter λ. The optimum value for λ must be found by experiment, although simulations suggest that the results are not strongly dependent on the precise value chosen.

This technique can prevent overfitting without needing to limit the number of radial basis functions, and therefore allows all training data points to act as basis function centers. This may be particularly useful when the amount of training data is limited, or when the data are sparsely distributed in important regions of the input space. Furthermore, for many problems it is known that the desired mapping should

have certain smoothness properties, and this technique allows this to be imposed explicitly. Where appropriate, curvature-driven smoothing can easily be combined with techniques for restricting the number of basis functions.

Finally, the network can generate a large class of nonlinear multivariate mappings, while the learning algorithm corresponds to the solution of a linear problem and is therefore a fast one-step procedure.

References

Admoaitis, R. A. et al. 1990. Application of neural nets to systems identification and bifurcation analysis of real world experimental data. *Proceedings of International Conference on Neural Networks*, Lyons, France (in press).

Bishop, C. M. 1990. Curvature-driven smoothing in backpropagation neural networks. *Proceedings of the International Neural Network Conference*, Paris, Vol. 2, p. 749. Submitted to *Neural Networks*.

Broomhead, D. S., and Lowe, D. 1988 Multi-variable functional interpolation and adaptive networks. *Complex Syst.* **2**, 321.

Golub, G., and Kahan, W. 1965. Calculating the singular values and pseudo-inverse of a matrix. *J. SIAM Numerical Anal. Ser.* **B2**, 205.

Kohonen, T. 1988. *Self Organisation and Associative Memory*. Springer-Verlag, New York.

Michelli, C. A. 1986. Interpolation of scattered data: Distance matrices and conditionally positive definite functions. *Construct. Approx.* **2**, 11.

Poggio, T., and Girosi, F. 1990. Networks for approximation and learning. *Proc. IEEE* **78**(9), 1481.

Rumelhart, D. E., and McClelland, J. L. 1986. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Vol 1: *Foundations*. The MIT Press, Cambridge, MA.

Tikhonov, A. N., and Arsenin, V. Y. 1977. *Solutions of Ill-Posed Problems*. Wiley, New York.

Webb, A. R. 1991. Functional approximation by feed-forward networks: A least-squares approach to generalisation. RSRE Memorandum 4453, R.S.R.E., St Andrews Road, Malvern, Worcs., WR14 3PS, U.K.

Received 6 March 1991; accepted 18 April 1991.