



Introduction of the Radial Basis Function (RBF) Networks

Adrian G. Bors

adrian.bors@cs.york.ac.uk

Department of Computer Science

University of York

York, YO10 5DD, UK

Abstract

In this paper we provide a short overview of the Radial Basis Functions (RBF), their properties, the motivations behind their use and some of their applications. RBF's have been employed for functional approximation in time-series modeling and in pattern classification. They have been shown to implement the Bayesian rule and to model any continuous input-output mapping. RBF's are embedded in a two-layer neural network topology. We present the physical and statistical significance of the elements composing the network. We introduce a few RBF training algorithms and we show how RBF networks can be used in real applications.

1 Introduction

Radial Basis Functions emerged as a variant of artificial neural network in late 80's. However, their roots are entrenched in much older pattern recognition techniques as for example potential functions, clustering, functional approximation, spline interpolation and mixture models [1]. RBF's are embedded in a two layer neural network, where each hidden unit implements a radial activated function. The output units implement a weighted sum of hidden unit outputs. The input into an RBF network is nonlinear while the output is linear. Their excellent approximation capabilities have been studied in [2,3]. Due to their nonlinear approximation properties, RBF networks are able to model complex mappings, which perceptron neural networks can only model by means of multiple intermediary layers [4].

In order to use a Radial Basis Function Network we need to specify the hidden unit activation function, the number of processing units, a criterion for modeling a given task and a training algorithm for finding the parameters of the network. Finding the RBF weights is called network training. If we have at hand a set of input-output pairs, called training set, we optimize the network parameters in order to fit the network outputs to the given inputs. The fit is evaluated by means of a cost function, usually assumed to be the mean square error. After training, the RBF network can be used with data whose underlying statistics is similar to that of the training set. On-line training algorithms adapt the network parameters to the changing data statistics [5,6]. RBF networks have been successfully applied to a large diversity of applications including interpolation [7,8], chaotic time-series modeling [5,9], system identification, control engineering [10], electronic device parameter modeling, channel equalization [4,11,12], speech recognition [11,13], image restoration [14], shape-from-shading [15], 3-D object modeling [8,16], motion estimation and moving object segmentation [17], data fusion [18], etc.

This paper is structured as follows: in Section 2 we explain the network topology, in Section 3 we provide the RBF properties, while in Section 4 we describe some training algorithms for RBF networks. Some experimental results when applying RBF networks are provided in Section 5, while the conclusions of this study are drawn in Section 6.

2 Network topology

Radial basis functions are embedded into a two-layer feed-forward neural network. Such a network is characterized by a set of inputs and a set of outputs. In between the inputs and outputs there is a layer of processing units called hidden units. Each of them implements a radial basis function. The way in which the network is used for data modeling is different when approximating time-series and in pattern classification. In the first case, the network inputs represent data samples at certain past time-laps, while the network has only one output representing a signal value as shown in Fig. 1. In a pattern classification application the inputs represent feature entries, while each output corresponds to a class as shown in Fig 2. The hidden units correspond to subclasses in this case [11].

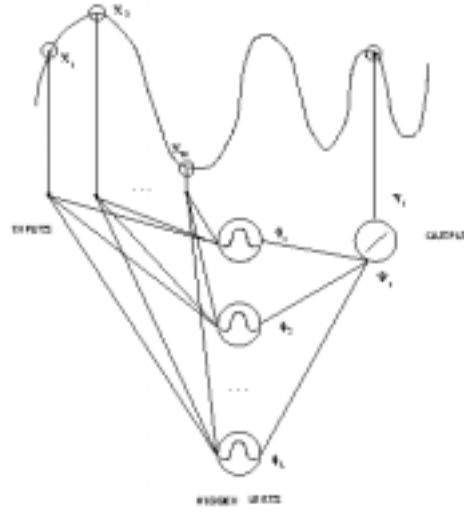


Figure 1. RBF network in time series modeling.

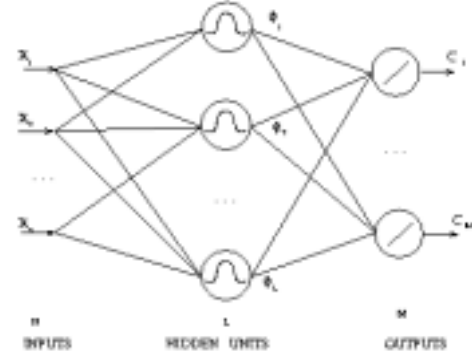


Figure 2. RBF network in pattern classification.

Various functions have been tested as activation functions for RBF networks [8,12]. In time-series modeling the most used activation function is the thin-plate spline [12]. In pattern classification applications the Gaussian function is preferred [3,5,6,10,11]. Mixtures of Gaussians have been considered in various scientific fields. The Gaussian activation function for RBF networks is given by:

$$\phi_j(\mathbf{X}) = \exp\left[-(\mathbf{X} - \boldsymbol{\mu}_j)^T \boldsymbol{\Sigma}_j^{-1} (\mathbf{X} - \boldsymbol{\mu}_j)\right] \quad (1)$$

for $j = 1, \dots, L$, where $\mathbf{X}$ is the input feature vector, L is the number of hidden units, $\boldsymbol{\mu}_j$ and $\boldsymbol{\Sigma}_j$ are the mean and the covariance matrix of the j th Gaussian function. In certain approaches a polynomial term is added to the expression (1) as in [7,9] while in others the Gaussian function is normalized to the sum of all the Gaussian components as in the Gaussian-mixtures estimation [14]. Geometrically, a radial basis function represents a bump in the multidimensional space, whose dimension is given by the number of entries. The mean vector $\boldsymbol{\mu}_j$ represent the location, while $\boldsymbol{\Sigma}_j$ models the shape of the activation function.

Statistically, an activation function models a probability density function where $\boldsymbol{\mu}_j$ and $\boldsymbol{\Sigma}_j$ represent the first and second order statistics.

The output layer implements a weighted sum of hidden-unit outputs:

$$\psi_k(\mathbf{X}) = \sum_{j=1}^L \lambda_{jk} \varphi_j(\mathbf{X})$$

for $k = 1, \dots, M$ where λ_{jk} are the output weights, each corresponding to the connection between a hidden unit and an output unit and M represent the number of output units. The weights λ_{jk} show the contribution of a hidden unit to the respective output unit. In a classification problem if $\lambda_{jk} > 0$ the activation field of the hidden unit j is contained in the activation field of the output unit k .

In pattern classification applications, the output of the radial basis function is limited to the interval (0,1) by a sigmoidal function:

$$Y_k(\mathbf{X}) = \frac{1}{1 + \exp[-\psi_k(\mathbf{X})]}$$

for $k = 1, \dots, M$.

RBF networks have been implemented in parallel hardware using systolic arrays [19].

3 Properties

The RBF's are characterized by their localization (center) and by an activation hypersurface. In the case of Gaussian functions these are modeled by the two parameters $\boldsymbol{\mu}_j$ and $\boldsymbol{\Sigma}_j$. The hypersurface is a hypersphere in the case when the covariance matrix is diagonal and has the diagonal elements equal, and is a hyperellipsoid in the general case. In the general case of a hyperellipsoid, the activation function influence decreases according to the Mahalanobis distance from the center. This means that data samples located at a large Mahalanobis distance from the RBF center will fail to activate that basis function. The maximum activation is achieved when the data sample coincides with the mean vector. Gaussian basis functions are quasi-orthogonal. The product of two basis functions, whose centers are far away from each other with respect to their spreads, is almost zero.

The RBF's can be thought of as potential functions. Hidden units whose output weights λ_{jk} have the same sign for a certain output unit, have their activation fields joined together in the same way as electrical charges of the same sign form electrical fields. For hidden units with output weights of different sign, their activation fields will correspond to the electrical field of opposite sign charges. Other similarities are with windowing functions [8,10] and with clustering [5,6].

RBF's properties made them attractive for interpolation and functional modeling. As a direct consequence, RBF's have been employed to model probability density functions. RBF networks have been shown to implement the Bayesian rule [3,11].

4 Training algorithms

By means of training, the neural network models the underlying function of a certain mapping. In order to model such a mapping we have to find the network weights and topology. There are two categories of training algorithms: supervised and unsupervised. RBF networks are used mainly in supervised applications. In a supervised application, we are provided with a set of data samples called training set for which the corresponding network

outputs are known. In this case the network parameters are found such that they minimize a cost function:

$$\min \sum_{i=1}^Q (\mathbf{Y}_k(\mathbf{X}_i) - \mathbf{F}_k(\mathbf{X}_i))^T (\mathbf{Y}_k(\mathbf{X}_i) - \mathbf{F}_k(\mathbf{X}_i)) \quad (2)$$

where Q is the total number of vectors from the training set, $\mathbf{Y}_k(\mathbf{X}_i)$ denotes the RBF output vector and $\mathbf{F}_k(\mathbf{X}_i)$ represents the output vector associated with the a data sample $\mathbf{X}_i$ from the training set. In unsupervised training the output assignment is not available for the given set.

A large variety of training algorithms has been tested for training RBF networks. In the initial approaches, to each data sample was assigned a basis function. This solution proved to be expensive in terms of memory requirement and in the number of parameters. On the other hand, exact fit to the training data may cause bad generalization. Other approaches choose randomly or assumed known the hidden unit weights and calculate the output weights λ_{jk} by solving a system of equations whose solution is given in the training set [7]. The matrix inversion required in this approach is computationally expensive and could cause numerical problems in certain situations (when the matrix is singular). In [8,10] the radial basis function centers are uniformly distributed in the data space. The function to be modeled is obtained by interpolation. In [13] less basis functions than given data samples are used. A least squares solution that minimizes the interpolation error is proposed.

Orthogonal least squares using Gram-Schmidt algorithm is proposed in [12,20]. An adaptive training algorithm for minimizing a given cost function is a gradient descend algorithm. Backpropagation adapts iteratively the network weights considering the derivatives of the cost function (2) with respect to those weights [3,4,11,21]. Expectation-maximization algorithm using a gradient descent algorithm for modeling the input-output distributions is employed in [14]. Backpropagation algorithm may require several iterations and can get stuck into a local minima of the cost function (2). Clustering algorithms as k -means [1], or learning vector quantization [22] have been employed for finding the hidden unit parameters in [5]. The centers of the radial basis functions are initialized randomly. This algorithm is on-line and its first stage is unsupervised. For a given data sample $\mathbf{X}_i$, the algorithm adapts its closest center:

$$\|\mathbf{X}_i - \hat{\boldsymbol{\mu}}_j\| = \min_{k=1}^L \|\mathbf{X}_i - \hat{\boldsymbol{\mu}}_k\|$$

The Euclidean distance denoted by $\|\cdot\|$ can be replaced with the Mahalanobis distance [6]. In this situation we use the covariance matrix in the computation of the distance. In order to avoid the singularity of the covariance matrix a sufficient large number of data samples must be considered for the covariance matrix calculation used in the Mahalanobis distance. The center is updated as follows:

$$\hat{\boldsymbol{\mu}}_j = \hat{\boldsymbol{\mu}}_j + \eta(\mathbf{X}_i - \hat{\boldsymbol{\mu}}_j)$$

where η is the training rate. For a minimal output variance, the training rate is equal to the inverse of the total number of data samples associated to that hidden unit. In this case the center corresponds to the classical first order statistical estimation. Similarly, second order statistical estimation can be employed for the covariance matrix. The output weights are evaluated in a second stage by means of Least Mean Square estimation. Outliers and data overlapping may cause bias in the parameter estimation. Algorithms using robust statistics as Median RBF [6,17] and Alpha-trimmed Mean RBF [16] have been employed for hidden unit estimation. A set of generator functions is proposed in [21] for hidden unit activation function selection. Stochastic selection has been considered in [23] for the radial basis functions.

As we have observed in Section 2, RBF network topology is determined by the number of hidden units. Various procedures have been employed for finding a suitable network

topology. Usually, topology adaptive approaches employ an additional regularization term to the cost function (2) depending on the number of hidden units. Criteria such as Akaike [12] or Minimum Description Length can be used in this case. Other approaches employ cluster merging [20] or splitting [11].

5 Experimental results

In the following we show some of the capabilities of the RBF network. We implement an RBF classifier on a set of artificial distributions. We considered four two-dimensional (2-D) Gaussian clusters grouped in two classes. The distributions are shown in Fig. 3.

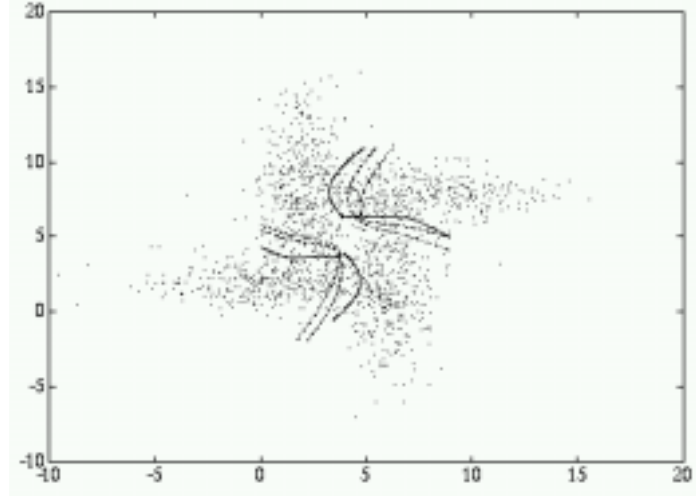


Figure 3. A set of four 2-D overlapping Gaussian distributions.

We can observe from Fig. 3 that the distributions are overlapping which creates an amount of mixture data in each distribution coming from the other distributions. We mark the optimal boundaries between the classes with continuous line, while the boundary found by a learning vector quantization based training algorithm for RBF networks is represented by dot-dash line. Median RBF (MRBF) [6] is displayed with dashed line. We can observe that even though not optimal, the second algorithm provide a better approximation of the boundary between the two classes. The topology of the network has two inputs and two outputs in both cases. The number of hidden units is 4 in the first case and 3 in the second.

In another example we model a composed shape made up of 6 overlapping three-dimensional (3-D) spheres using an RBF network. The shape to be modeled is shown in Fig. 4. In this case the network has three inputs from three coordinates, six hidden units and one output. In Fig. 5 is shown the shape modeled by means of an RBF network trained using a learning vector quantization algorithm while in Figs. 6 and 7 are the shapes resulted after using MRBF [6] and Alpha-trimmed mean RBF [16]. We have also added uniformly distributed noise to

Table I. Comparison among three RBF training algorithms for modeling a 3-D shape.

RBF Training Algorithms	Noise-free Model		Noisy Model	
	Bias in Center Estimation	Modeling Error (%)	Bias in Center Estimation	Modeling Error (%)
RBF	2.02	11.59	5.38	59.03
MRBF	3.64	27.03	4.30	32.32
Alpha-trimmed mean RBF	1.40	5.99	2.93	20.37

the 3-D shape from Fig. 4. In order to assess the modeling we have considered two error measures. The first error measure estimates the bias in sphere center estimation. The second error measure computes the error in shape estimation by comparing the given shape and the shape implemented by the network. The comparative results are displayed in Table I. We can observe from this table that algorithms using robust statistics provide better parameter estimation than classical RBF network estimation.

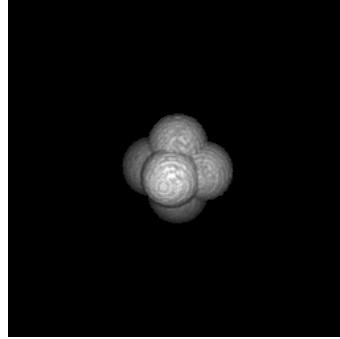


Figure 4. Initial shape representation using 6 spheres.

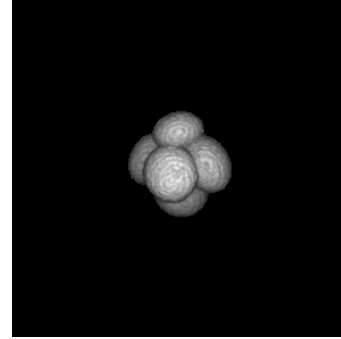


Figure 5. The shape learned by an RBF network.

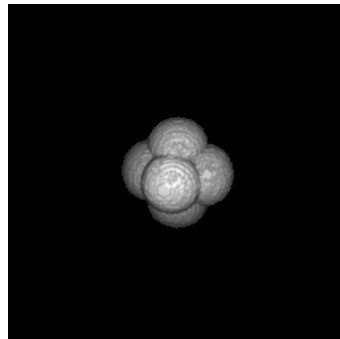


Figure 6. Modeling using Alpha-trimmed mean RBF network.

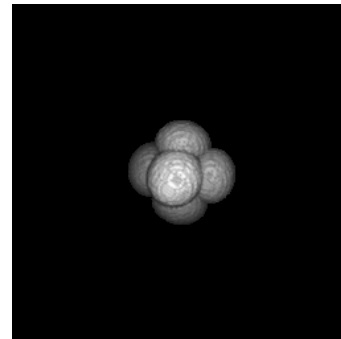


Figure 7. Modeling using MRBF.

6 Conclusions

In this study we provide an introduction to the Radial Basis Function Networks. RBF's have very attractive properties such as localization, functional approximation, interpolation, cluster modeling and quasi-orthogonality. These properties made them attractive in many applications. Very different fields such as: telecommunications, signal and image processing, control engineering and computer vision used them successfully for various tasks. We provide some of their properties and a few training algorithms. We present some examples when applying RBF network to modeling and classifying artificial data.

References

- [1] Tou, J. T., Gonzalez, R. C. (1974) *Pattern Recognition*. Reading, MA: Addison-Wesley.
- [2] Park, J., Sandberg, J. W. (1991), "Universal approximation using radial basis functions network," *Neural Computation*, vol. 3, pp. 246-257.

- [3] Poggio, T., Girosi, F., (1990) "Networks for approximation and learning," *Proc. IEEE*, vol. 78, no. 9, pp. 1481-1497.
- [4] Haykin, S. (1994) *Neural Networks: A comprehensive Foundation*. Upper Saddle River, NJ: Prentice Hall.
- [5] Moody, J., (1989) "Fast learning in networks of locally-tuned processing units," *Neural Computation*, vol. 1, pp. 281-294.
- [6] Bors, A.G., Pitas, I., (1996) "Median radial basis functions neural network," *IEEE Trans. on Neural Networks*, vol. 7, no. 6, pp. 1351-1364. Casdagli, M. (1989) "Nonlinear prediction of chaotic time series," *Physica D*, vol. 35, pp. 335-356.
- [7] Broomhead, D.S., Lowe, D. (1988) "Multivariable functional interpolation and adaptive networks," *Complex Systems*, vol. 2, pp. 321-355.
- [8] Matej, S., Lewitt, R.M., (1996) "Practical considerations for 3-D image reconstruction using spherically symmetric volume elements," *IEEE Trans. on Medical Imaging*, vol. 15, no. 1, pp. 68-78.
- [9] Casdagli, M., (1989) "Nonlinear prediction of chaotic time series," *Physica D*, vol. 35, pp. 335-356.
- [10] Sanner, R. M., Slotine, J.-J. E., (1994) "Gaussian networks for direct adaptive control," *IEEE Trans. On Neural Networks*, vol. 3, no. 6, pp. 837-863.
- [11] Bors, A. G., Gabbouj, G., (1994) "Minimal topology for a radial basis function neural network for pattern classification," *Digital Signal Processing: a review journal*, vol. 4, no. 3, pp. 173-188.
- [12] Chen, S., Cowan, C. F. N., Grant, P. M. (1991) "Orthogonal least squares learning algorithm for radial basis function networks," *IEEE Trans. On Neural Networks*, vol. 2, no. 2, pp. 302-309.
- [13] Niranjana, M., Fallside, F., (1990) "Neural networks and radial basis functions in classifying static speech patterns," *Computer Speech and Language*, vol. 4, pp. 275-289.
- [14] Cha, I., Kassam, S. A., (1996) "RBFN restoration of nonlinearly degraded images," *IEEE Trans. on Image Processing*, vol. 5, no. 6, pp. 964-975.
- [15] Wei, G.-Q., Hirzinger, G., (1997) "Parametric shape-from-shading by radial basis functions," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 19, no. 4, pp. 353-365.
- [16] Bors, A.G., Pitas, I., (1999) "Object classification in 3-D images using alpha-trimmed mean radial basis function network," *IEEE Trans. on Image Processing*, vol. 8, no. 12, pp. 1744-1756.
- [17] Bors, A.G., Pitas, I., (1998) "Optical flow estimation and moving object segmentation based on median radial basis function network," *IEEE Trans. on Image Processing*, vol. 7, no. 5, pp. 693-702.
- [18] Chatzis, V., Bors, A. G., Pitas, I., (1999) "Multimodal decision-level fusion for person authentication," *IEEE Trans. on Systems, Man, and Cybernetics, part A: Systems and Humans*, vol. 29, no. 6, pp. 674-680.
- [19] Broomhead, D. S., Jones, R., McWhirter, J. G., Shepherd, T. J., (1990) "Systolic array for nonlinear multidimensional interpolation using radial basis functions," *Electronics Letters*, vol. 26, no. 1, pp. 7-9.
- [20] Musavi, M.T., Ahmed, W., Chan, K.H., Faris, K.B., Hummels, D.M., (1992) "On the training of radial basis function classifiers," *Neural Networks*, vol. 5, pp. 595-603.
- [21] Karayiannis, N. B. (1999) "Reformulated radial basis neural networks trained by gradient descent," *IEEE Trans. on Neural Networks*, vol. 10, no. 3, pp. 657-671.
- [22] Kohonen, T. K., (1989) *Self-organization and associative memory*. Berlin: Springer-Verlag.
- [23] Igel'nik, B., Y.-H. Pao, (1995) "Stochastic choice of radial basis functions in adaptive function approximation and the functional-link net," *IEEE Trans. on Neural Networks*, vol. 6, no. 6, pp. 1320-1329.