

Radial Basis Function Networks: Introduction

Introduction to Neural Networks : Lecture 12

© John A. Bullinaria, 2004

1. Introduction to Radial Basis Functions
2. Exact Interpolation
3. Common Radial Basis Functions
4. Radial Basis Function (RBF) Networks
5. Problems with Exact Interpolation Networks
6. Improving RBF Networks
7. The Improved RBF Network

Introduction to Radial Basis Functions

The idea of *Radial Basis Function (RBF) Networks* derives from the theory of function approximation. We have already seen how Multi-Layer Perceptron (MLP) networks with a hidden layer of sigmoidal units can learn to approximate functions. RBF Networks take a slightly different approach. Their main features are:

1. They are two-layer feed-forward networks.
2. The hidden nodes implement a set of radial basis functions (e.g. Gaussian functions).
3. The output nodes implement linear summation functions as in an MLP.
4. The network training is divided into two stages: first the weights from the input to hidden layer are determined, and then the weights from the hidden to output layer.
5. The training/learning is very fast.
6. The networks are very good at interpolation.

We'll spend the next three lectures studying the details...

Exact Interpolation

The *exact interpolation* of a set of N data points in a multi-dimensional space requires every one of the D dimensional input vectors $\mathbf{x}^p = \{x_i^p : i = 1, \dots, D\}$ to be mapped onto the corresponding target output t^p . The goal is to find a function $f(\mathbf{x})$ such that

$$f(\mathbf{x}^p) = t^p \quad \forall p = 1, \dots, N$$

The radial basis function approach introduces a set of N *basis functions*, one for each data point, which take the form $\phi(\|\mathbf{x} - \mathbf{x}^p\|)$ where $\phi(\cdot)$ is some non-linear function whose form will be discussed shortly. Thus the p th such function depends on the distance $\|\mathbf{x} - \mathbf{x}^p\|$, usually taken to be Euclidean, between \mathbf{x} and \mathbf{x}^p . The output of the mapping is then taken to be a linear combination of the basis functions, i.e.

$$f(\mathbf{x}) = \sum_{p=1}^N w_p \phi(\|\mathbf{x} - \mathbf{x}^p\|)$$

The idea is to find the “weights” w_p such that the function goes through the data points.

Determining the Weights

It is easy to determine equations for the weights by combining the above equations:

$$f(\mathbf{x}^q) = \sum_{p=1}^N w_p \phi(\|\mathbf{x}^q - \mathbf{x}^p\|) = t^q$$

We can write this in matrix form by defining the vectors $\mathbf{t} = \{t^p\}$ and $\mathbf{w} = \{w_p\}$, and the matrix $\Phi = \{\Phi_{pq} = \phi(\|\mathbf{x}^q - \mathbf{x}^p\|)\}$. This simplifies the equation to $\Phi \mathbf{w} = \mathbf{t}$. Then, provided the inverse of Φ exists, we can use any standard matrix inversion techniques to give

$$\mathbf{w} = \Phi^{-1} \mathbf{t}$$

It can be shown that, for a large class of basis functions $\phi(\cdot)$, the matrix Φ is indeed non-singular (and hence invertible) providing the data points are distinct.

Once we have the weights, we have a function $f(\mathbf{x})$ that represents a continuous differentiable surface that passes exactly through each data point.

Commonly Used Radial Basis Functions

A range of theoretical and empirical studies have indicated that many properties of the interpolating function are relatively insensitive to the precise form of the basis functions $\phi(r)$. Some of the most commonly used basis functions are:

1. Gaussian Functions:

$$\phi(r) = \exp\left(-\frac{r^2}{2\sigma^2}\right) \quad \text{width parameter } \sigma > 0$$

2. Multi-Quadric Functions:

$$\phi(r) = (r^2 + \sigma^2)^{1/2} \quad \text{parameter } \sigma > 0$$

3. Generalized Multi-Quadric Functions:

$$\phi(r) = (r^2 + \sigma^2)^\beta \quad \text{parameters } \sigma > 0, 1 > \beta > 0$$

4. Inverse Multi-Quadric Functions:

$$\phi(r) = (r^2 + \sigma^2)^{-1/2} \quad \text{parameter } \sigma > 0$$

5. Generalized Inverse Multi-Quadric Functions:

$$\phi(r) = (r^2 + \sigma^2)^{-\alpha} \quad \text{parameters } \sigma > 0, \alpha > 0$$

6. Thin Plate Spline Function:

$$\phi(r) = r^2 \ln(r)$$

7. Cubic Function:

$$\phi(r) = r^3$$

8. Linear Function:

$$\phi(r) = r$$

Properties of the Radial Basis Functions

The Gaussian and Inverse Multi-Quadric Functions are *localised* in the sense that

$$\phi(r) \rightarrow 0 \quad \text{as} \quad |r| \rightarrow \infty$$

but this is not strictly necessary. All the other functions above have the property

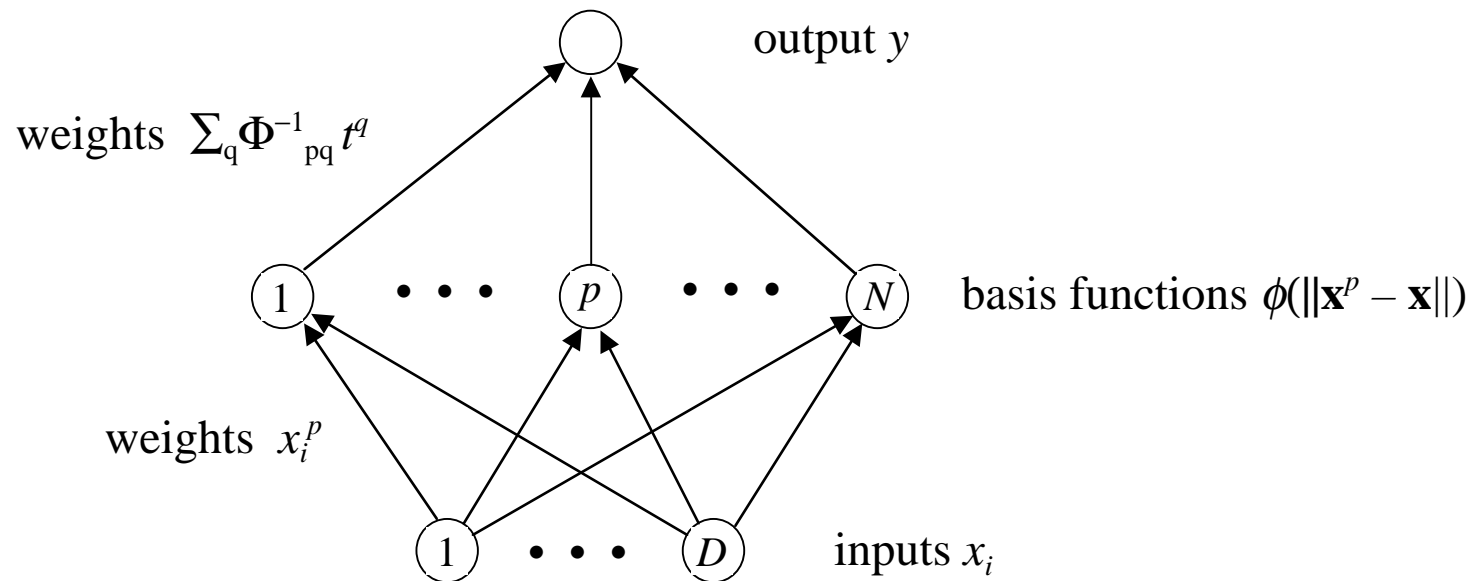
$$\phi(r) \rightarrow \infty \quad \text{as} \quad |r| \rightarrow \infty$$

Note that even the Linear Function $\phi(r) = r = \|\mathbf{x} - \mathbf{x}^p\|$ is still non-linear in the components of \mathbf{x} . In one dimension, this leads to a piecewise-linear interpolating function which performs the simplest form of exact interpolation.

For neural network mappings, there are good reasons for preferring localised basis functions. We shall focus our attention on *Gaussian basis functions* since, as well as being localised, they have a number of other useful analytic properties. We can also see intuitively how to set their widths σ and build up function approximations with them.

Radial Basis Function Networks

You might think that what we have just described isn't really a neural network. And a lot of people would agree with you! However, we can see how to make it look like one:



Note that the N training patterns $\{ x_i^p, t^p \}$ determine the weights directly. The hidden layer to output weights multiply the hidden unit activations in the conventional manner, but the input to hidden layer weights are used in a very different fashion.

Problems with Exact Interpolation

We have seen how we can set up RBF networks that perform exact interpolation, but there are two serious problems with these exact interpolation networks:

1. They perform poorly with noisy data

As we have already seen for Multi-Layer Perceptrons (MLPs), we do not usually want the network's outputs to pass through all the data points when the data is noisy, because that will be a highly oscillatory function that will not provide good generalization.

2. They are not computationally efficient

The network requires one hidden unit (i.e. one basis function) for each training data pattern, and so for large data sets the network will become very costly to evaluate.

With MLPs we can improve generalization by using more training data – the opposite happens in RBF networks, and they take longer to compute as well.

Improving RBF Networks

We can take the basic structure of the RBF networks that perform exact interpolation and improve upon them in a number of ways:

1. The number M of basis functions (hidden units) need not equal the number N of training data points. In general it is better to have M much less than N .
2. The centres of the basis functions do not need to be defined as the training data input vectors. They can instead be determined by a training algorithm.
3. The basis functions need not all have the same width parameter σ . These can also be determined by a training algorithm.
4. We can introduce bias parameters into the linear sum of activations at the output layer. These will compensate for the difference between the average value over the data set of the basis function activations and the corresponding average value of the targets.

Of course, these will make analysing and optimising the network much more difficult.

The Improved RBF Network

When these changes are made to the exact interpolation formula, and we allow the possibility of more than one output unit, we arrive at the RBF network mapping

$$y_k(\mathbf{x}) = w_{k0} + \sum_{j=1}^M w_{kj} \phi_j(\mathbf{x})$$

which we can simplify by introducing an extra basis function $\phi_0 = 1$ to give

$$y_k(\mathbf{x}) = \sum_{j=0}^M w_{kj} \phi_j(\mathbf{x})$$

For the case of Gaussian basis functions we have

$$\phi_j(\mathbf{x}) = \exp\left(-\frac{\|\mathbf{x} - \boldsymbol{\mu}_j\|^2}{2\sigma_j^2}\right)$$

in which we have $M \times D$ basis centres $\{\boldsymbol{\mu}_j\}$ and M widths $\{\sigma_j\}$. Next lecture we shall see how to determine all the network parameters $\{w_{kj}, \boldsymbol{\mu}_j, \sigma_j\}$.

Overview and Reading

1. We began by outlining the basic properties of RBF networks.
2. We then looked at the idea of exact interpolation using RBFs, and went through a number of common RBFs and their important properties.
3. We then saw how to set up an RBF network for exact interpolation and noted two serious problems with it.
4. We ended by formulating a more useful form of RBF network.

Reading

1. Bishop: Sections 5.1, 5.2, 5.3, 5.4
2. Haykin: Sections 5.1, 5.2, 5.3, 5.4
3. Gurney: Section 10.4
4. Callan: Section 2.6
5. Hertz, Krogh & Palmer: Section 9.7