ABSTRACT

This chapter presents an algorithm to train radial basis function neural networks (RBFNs) in a semi-online manner. It employs the online, evolving clustering algorithm of Kasabov and Song (2002) in the unsupervised training part of the RBFN and the ordinary least squares estimation technique for the supervised training part. Its effectiveness is demonstrated on two problems related to bankruptcy prediction in financial engineering. In all the cases, 10-fold cross validation was performed. The present algorithm, implemented in two variants, yielded more sensitivity compared to the multi-layer perceptron trained by backpropagation (MLP) algorithm over all the problems studied. Based on the results, it can be inferred that the semi-online RBFN without linear terms is better than other neural network techniques. By taking the Area Under the ROC curve (AUC) as the performance metric, the proposed algorithms—semi-online RBFN with and without linear terms—are compared with classifiers such as ANFIS, TreeNet, SVM, MLP, Linear RBF, RSES, and Orthogonal RBF. Out of them TreeNet outperformed both the variants of the semi-online RBFN in both data sets considered here.
INTRODUCTION

Artificial neural networks (ANNs) have been applied in applications involving classification, function approximation, optimization, and control. It is well known that the two most popular architectures of ANN—multi-layer perceptron (MLP) and radial basis function network (RBFN)—are universal function approximators. RBFN and MLP can be used for a wide range of applications primarily because they can approximate any function under mild conditions. MLP is trained by supervised learning. On the contrary, the training of RBFN takes place in a hybrid manner containing both unsupervised and supervised schemes. Unsupervised training is less approximate and hence relatively fast. Moreover, the supervised part of the learning consists of solving a linear problem, which is therefore fast, with the additional benefit of avoiding the problem of local minima usually encountered in training MLP. Hence, the training of RBFN is faster than that of MLP. RBFN has just two layers of parameters (centers, widths, and weights) and each layer can be determined sequentially (Benoudjit & Verleysen, 2003).

An RBF network has two layers. Consider an unknown function \( f(X) : \mathbb{R}^d \rightarrow \mathbb{R} \). In a regression context, RBFN approximates \( f(X) \) by a weighted sum of d-dimensional radial activation functions (plus linear and independent terms). The radial basis functions are centered on well-positioned data points, called centroids; the centroids can be regarded as the nodes of the hidden layer. The positions of the centroids are obtained by an unsupervised learning rule. The network weights between the radial basis function layer and the output layer are estimated using ordinary least squares technique. Suppose we want to approximate the function \( f(X) \) with a set of \( M \) radial basis functions \( \phi_j(X) \), centered on the centroids \( C_j \) and defined by Benoudjit and Verleysen (2003):

\[
\phi_j(X) : \mathbb{R}^d \rightarrow \mathbb{R} : \phi_j(X) = \phi(\|X - C_j\|)
\]

where \( \|\cdot\| \) denotes the Euclidean distance, \( C_j \in \mathbb{R}^d \) and \( 1 \leq j \leq M \).

The approximation of the function \( f(X) \) may be expressed as a linear combination of the radial basis functions:

\[
\sum_{j=1}^{M} \lambda_j \phi_j(\|X - C_j\|) + \sum_{i=1}^{d} a_i x_i + b
\]

where \( \lambda_j \) are weight factors, and \( a_i, b \) are the weights for the linear and independent terms respectively.

A typical choice for the radial basis functions is a set of multi-dimensional Gaussian kernels:

\[
\phi_j(\|X - C_j\|) = \exp\left(-\frac{1}{2} \|X - C_j\|^2 \right)
\]

Moody and Darken (1989) proposed, as an unsupervised part of the algorithm, the k-means clustering algorithm to find the location of the centroids \( C_j \). Once the basis function parameters are determined, the transformation between the input data and the corresponding outputs of the hidden units is fixed. Then, the supervised learning part of the algorithm commences where the weights connecting the nodes in the kernel layer and the nodes in the output layer are estimated using the linear least squares technique. Accordingly, the minimization of the average mean square error yields the least square solution for the weights.

\[
\lambda = \phi^T y = (\phi^T \phi)^{-1} \phi^T y,
\]

where \( \lambda, y \) are the row vectors of weight factors \( \lambda_j \) and training data outputs \( y_j \), \( \phi \) is \( N \times M \) matrix of \( \phi_j \) exp(\( -\|X - C_j\|^2 / 2 \sigma^2 \)) values, and \( \phi^T = (\phi^T \phi)^{-1} \phi^T \) denotes the pseudo-inverse of \( \phi \).

Review of Work Done in Improving RBF Network

We now review earlier works where online training algorithms for RBFN were suggested. Fung, Billings, and Luo (1996) derived a new recursive supervised training algorithm, which combines