Adaptive receptive fields for radial basis functions

Mutaz M. Jafar*, Ali Zilouchian

Department of Electrical Engineering, Florida Atlantic University, Boca Raton, FL 33431, USA
Tel. +1 (561) 297-3468; Fax +1 (561) 297-2336

Received 20 September 2000; accepted 5 October 2000

Abstract

We propose a network architecture based on adaptive receptive fields and a learning algorithm that combines both supervised learning of centers and unsupervised learning of output layer weights. This algorithm causes each group of radial basis functions to adapt to regions of the clustered input space. Networks produced by this algorithm appear to have better generalization performance on prediction of non-linear input-output mappings than corresponding back-propagation algorithms and requires a fraction of the number of connection weights required by fixed center radial basis. For a test problem of predicting product quality of a reverse osmosis desalination plant, the network learns much faster than a three-layer perceptron trained with back-propagation, but requires additional computational burden.

Keywords: Reverse osmosis; Neural network; Prediction; Radial basis function network

1. Introduction

Finding an approximate model is an issue, which engineers are faced with when dealing with complex, large-scale industrial applications. The objective of modelling is to closely approximate input-output mapping. However, most physical systems have to be modelled separately due to complexity and presence of non-linear characteristics [1]. Neural networks (NN) have been used in a large number of applications and have proven to be effective in performing complex functions in various fields. These include pattern recognition, classification, control systems, and prediction of critical parameters in non-linear industrial processes [2,3].

Adaptation or learning is a major focus of neural net research that provides a degree of robustness to the NN model. In predictive modeling, the goal is to map a set of input patterns onto a set of output patterns. NN accomplishes this task by learning from a series of data sets to the system, then applies what has

* Corresponding author.
learned to approximate or predict the corresponding output [4,5].

The most successful modern network-learning model has been back-propagation. Based on Least Mean Square (LMS) algorithm, the back-propagation method has been widely used to solve a number of applications [6,7]. However, despite the practical success, back-propagation suffer from slow convergence. While optimization of learning rate and momentum coefficient parameters yields overall improvements on the networks, it is still inefficient and time consuming for real-time applications [8,9].

Radial Basis Function Networks (RBFN) with local overlapping receptive fields provides an alternative to sigmoid networks for predictive modeling. They perform excellent approximations for curve fitting problems, and can be trained easily and quickly. However, RBFN usually exhibits a slow response in the recall phase due to the large number of neurons associated in the second layer [10,11].

In the development of predictive model for desalination plants, application of NN is essential due to non-linearity and complexity of interactions between operating variable [12]. In particular, prediction, and adaptation to preventive measures, of key parameters in Reverse Osmosis (RO) can decrease membrane degradation and increase the overall efficiency. Most of pervious work in the field of NN application to desalination processes has been focused on thermal desalination. This included work by Parenti et al. [13] and Selvaragi and Deshpande [14] who applied NN for identification of Multi-stage flash desalination. In [15] Al-Shayji used back-propagation in conjunction with statistical techniques in order to predict output variables of both thermal desalination (MSF), and RO plants.

This paper discusses two methods of NN-based predictions using LMS and RBFN algorithms. Two variants of each method will be considered. The second variant of RBFN is a newly developed learning algorithm based on redistribution of receptive fields using data clustering and histogram equalization. The supervised learning component adjusts the center of the grouped receptive fields to regions of the input space where data is more meaningful. Once the centers are determined, the width of the receptive fields and the weights of second layer neurons are then determined.

We shall demonstrate the superiority of this method of combined supervised and unsupervised learning by more accurately predicting water quality of two actual data sets of RO desalination plants with two different feed water intakes, ground- and seawater. Furthermore, in comparison to traditional RBFN, the learning algorithm provides a more efficient design with a fraction of the number of neurons required.

2. RBFN learning strategies

One of the advantages of RBF is the fact that linear weights associated with the output layer can be treated separately from the hidden layer neurons. As the hidden layer weights are adjusted through a non-linear optimization, output layer weights are adjusted through linear optimization. Methods of adjustment of centers $u_i$ and spread widths $\sigma_i$ of the radial basis are described next.

In the fixed center form of learning, centers of the RBFN are randomly selected at fixed points from the training data set. During training, centers and widths are kept at a constant value while the output weight vector is the only parameter that is changed during training. The value of $\sigma$ must ensure that the gaussian functions overlap but do not have too sharp or flat shape.

RBFN training with fixed centers and width of the receptive field could present a sampling error if the training set does not have a good representation of the original data or if the widths of the receptive fields do not sufficiently overlap [16]. In addition, RBFN uses exponentially decaying localized non-linearities to con-
struct non-linear approximations. This can also lead to increase in the number of neurons and connection weights $N$, a number that grows with the dimensionality of the space to be searched. It is therefore essential to limit the number of hidden layer neurons so that the RBFN would not exhibit a slow response in the recall phase of the approximation.

In unsupervised learning, the centers move in a self-organized way to the desired location, while the linear weights of the output layer are computed by supervised learning such as LMS algorithm. Moody and Darken [17] combined both self-organized and supervised learning using k-nearest-neighbor rule to determine the center locations whereas the average distance to the center determined the widths. This method used a small number of RBFN but required a large number of training samples. In [18], Nowlan used a soft-competitive algorithm among gaussian hidden units for placement of radial basis centers. The method performed well in the classification application, but required computations of mean vector of all hidden units.

In the third form of learning strategies, a supervised learning algorithm will determine the centers, widths and the weights of the output layer. Gradient decent algorithm can be used to minimize the error between the actual and the desired output over the training set. The error is defined as:

$$e = d_i - \sum_{i=1}^{N_i} w_i \phi(\|x - x_i\|)$$  \hspace{1cm} (1)

Wettschereck and Dietterich [19], proposed a supervised method of generalized radial basis function to determine center locations the as well as output weights. Whitehead and Choate [20] used genetic algorithm and space-filling curve to determine the centers and genetic encoding to determine the widths of RBFN.

Other statistical methods of supervised learning included work done by Lowe [16] who proposed a way to determine the centers based on standard deviation and Chen [21] who used a method of orthogonal least squares to determine $u_i$ while keeping $\sigma_i$ constant.

3. Adaptive receptive fields for redistribution of radial basis functions

RBFN accuracy and speed may be further improved by strategically placing appropriate centers and widths of the receptive fields. The redistribution of centers to locations where input training data is meaningful can lead to more efficient RBFN. This new method combines both supervised and unsupervised learning is presented herein. The method is based on clustering of input space vectors, and computing weights of chosen Euclidean distances. Histogram equalization [22], within each cluster, determines centers and widths that lie inside the generality of the convex hull of the input space and yields a radial basis are reasonably spread out over the input space.

The supervised part of the algorithm includes redistribution of the centers and widths of receptive fields over the input space and computation of weight and bias matrix. The unsupervised part of the algorithm includes computation of the output weight and bias matrix. Summary of the steps involved in the adaptive receptive field training is shown in Fig. 1.

4. Methodology

Clustering methods have been used extensively to organize and categorize data, and proved to be useful in model construction [23]. Basically, clustering partitions data set into subgroups that have similar input-output pairs. K-means and fuzzy C-means clustering are two of the most common methods that are frequently used with radial basis [11,24]. Specht [8] used an effective clustering method based on determination of radius of influence. Training data is first normalized to a value between $[-1,1]$. A
radius of influence (RI) is then specified, and the first point establishes a new cluster center at \( x_i \). Each vector \( x \) in the input space is considered one at a time and if a sample with distance \(|x - x_i|\) to the nearest cluster is \( > r \), the vector center becomes the center of the new cluster. If the distance of \(|x - x_i|\) of the sample is less than the distance to any other cluster center and is \( \leq r \) the vector is assigned to that cluster. This procedure performs clustering in non-iterative way and requires only one pass through the training set.

Histogram equalization is a method used in digital image enhancement techniques [25, 26]. Basically, histogram equalization stretches the contrast of an image by uniformly redistributing the gray values. For the selected \( k^{th} \) output running sums of the \( i^{th} \) input vector are evaluated by:

\[
y_k = \sum_{i=1}^{k} y_i
\] (2)

A plot of running sums is evaluated and the number of radial basis functions evenly divides the Y-axis. The values of the resultant input vector will be used to determine the new Euclidean distances over the input space. Fig. 2 shows a histogram of case study 1 and resultant redistributed radial basis is shown in Fig. 3.

Once the Euclidean distances are determined, the first layer weights are then computed. This involves solving \( N \) non-linear equation of the form,

\[
\|x - x_{\alpha}\| = \sqrt{\sum_{i=1}^{N} (x_i - x_{\alpha_i})^2}
\] (3)

Where \( \|x - x_{\alpha}\| \) is the desired Euclidean distance of the \( i^{th} \) cluster, \( N \) is the number of clustering
centers, \( x_i \) is the average mean of the input cluster, and \( x_{\alpha i} \) are the weights to be calculated. The method of Generalized Reduced Gradient Non-Linear Optimization (GRG2) Code developed by Ladson and Waren [27] is used for solving the set of non-linear equations. The solution to the minimization of the function \( y_k = \sum_{i=1}^{N} \| x - x_{\alpha i} \|^2 \) determines the set of weights \( x_{\alpha i} \).

The bias \( b_i \) of the hidden layer neurons allows the sensitivity of the neuron to be adjusted, this is determined by:

\[
b_i = \frac{\sqrt{-\log(0.5)}}{\sigma_i}
\]  

which results in a radial function that crosses 0.5 at weighted inputs of ± width of the receptive field \( \sigma_i \). The width of the receptive region can have a critical effect on the accuracy of the radial basis network. Proper assignment of the widths can uniformly cover the input space with receptive fields and allow the network to properly respond to new input vectors. In our method the spacing between the resultant centers are not uniform, therefore, the shape of the receptive field can be different for each hidden layer neuron. We considered in our simulation a uniform and a variable value of \( \sigma \) for each cluster. In all of the simulations carried out on different data sets, it was found that the choice of a single global value of \( \sigma \) gave better results than separate values for each cluster. The value of \( \sigma \) in our study was set as the average standard deviation of the input histogram.

In the unsupervised part of the learning algorithm, output layer weights \( W_j \) and bias \( b_j \) are determined by minimizing the sum-squared error defined by,

\[
[W_j b_j] \times [a^1] = T
\]

where \( a^1 \) is output of 1\(^{st} \) layer and \( T \) is the target vector.

5. Application to two prediction tasks

Permeate flux, system recovery, and total dissolved solids (TDS) are three of the most important factors that indicate the performance of the reverse osmosis (RO) system. As an example, a low permeate flow combined with high salt passage could indicate colloidal fouling, metal oxide fouling or membrane scaling. Low permeate flow combined with normal salt passage may indicate biological fouling of the membrane.

When a large number of elements are combined in an RO system with a complex series-parallel-series configuration and only inlet operating variables are known, system performance prediction becomes considerably more complex. Feed pressure and salt concentration for each element in series are changing. The extent of these changes now depends not only on input variables but also on interaction of several variables [28–29]. Fig. 4 shows effect of four variables on product quality for seawater desalination.

It is therefore essential for early detection of potential problems and proper adjustment of operating variables in such a way that fouling or scaling will not occur. Prediction of product quality is a key factor to decrease membrane degradation and the overall efficiency of the RO system.

6. Simulation results

In this section, we present our simulation results that compare adaptive receptive field to the backpropagation for predicting water quality of two different RO feed intakes. Both networks have normalized real-valued inputs and one
The backpropagation network has two internal layers with 28 and 13 sigmoidal neurons for the first layer and second layer, respectively. Initialized of the network was based on the technique of Nguyen and Widrow [30]. Throughout the testing and simulation of neural network parameters, Matlab-5 with Neural Networks Toolbox [31] was used on a set of real-time data. The available data was partitioned to two sets, training and test set. The training set is further partitioned into two sets: training and validation. The networks were then simulated with the selected training data set. Results of the cross-validation curves are shown in Fig. 5.

The results, in general, are satisfactory since the test and validation errors have similar characteristics. No over- or under-fitting is visible and therefore, we conclude that the training set and parameters chosen were adequate. \( L_2 \) and \( L_\infty \) norms error criteria described in [32,33] were used for comparison and error analysis. Results of backpropagation and that of the adaptive RBFN are shown Figs. 6 and 7. Summary of comparison between the four algorithms is shown Tables 1 and 2.

7. Conclusions

Simulation results shown in previous section provide strong evidence that redistributed centers of RBFN receptive fields leads to better prediction.
Fig. 5. LMS and corresponding validation curves for selected values of learning rate (0.05 top curves, 0.1 middle curves, 0.5 bottom curves).

Fig. 6. Actual and predicted permeate flow (a) back-propagation (b) Adaptive RBFN.

Performance than other LMS and fixed center RBFN. By employing the supervised learning of the centers and unsupervised learning of the output-layer weights we can vastly improve the performance of the RBFN. This method resulted in more efficient design of the RBFN network by re-distributing the centers to locations where input training data is meaningful and by reducing the number of hidden neurons.

Training all layers of the LMS networks by back-propagation of the output error signal resulted in overall slow convergence. On the other hand, RBFN outperformed LMS algorithms with optimized two hidden layers trained with the same data vectors. However, additional computation was required for determining the weights matrix.

NN can be very effective in predicting and optimizing non-linear behaviour problems associated with operational variables of a large-scale reverse osmosis desalination plants. NN algorithms achieved an overall good generalization for predictions of permeate flow of RO plants. The prediction of permeate TDS resulted in overall lower accuracy. This is due to the fact of the high non-linearity of permeate TDS and dependence on other non-parameterized factors such as membrane oxidation, scaling and concentration polarization.

Adaptive RBFN approach can be very useful for real-time predictions due to its high efficiency over the conventional back-propagation approach and is likely to be useful in real-time applications with non-linear behaviour of the processing task.
Table 1
Summary of error norms (permeate prediction)

<table>
<thead>
<tr>
<th>Method</th>
<th>$\varepsilon_\infty$</th>
<th>$\varepsilon_2$</th>
<th>Total neurons</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Back-propagation</td>
<td>2.084</td>
<td>0.876</td>
<td>41</td>
<td></td>
</tr>
<tr>
<td>Fixed centers RBFN</td>
<td>2.36</td>
<td>0.593</td>
<td>39</td>
<td>$\sigma=0.35$</td>
</tr>
<tr>
<td>Adaptive RBFN</td>
<td>1.73</td>
<td>0.382</td>
<td>9</td>
<td>RI=0.3; $\sigma=0.92$</td>
</tr>
</tbody>
</table>

Table 2
Summary of error norms (TDS prediction)

<table>
<thead>
<tr>
<th>Method</th>
<th>$\varepsilon_\infty$</th>
<th>$\varepsilon_2$</th>
<th>Total neurons</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Back-propagation</td>
<td>3.41</td>
<td>1.87</td>
<td>41</td>
<td></td>
</tr>
<tr>
<td>Fixed centers RBFN</td>
<td>3.67</td>
<td>1.63</td>
<td>39</td>
<td>$\sigma=0.42$</td>
</tr>
<tr>
<td>Adaptive RBFN</td>
<td>2.32</td>
<td>0.95</td>
<td>4</td>
<td>RI=0.38; $\sigma=0.84$</td>
</tr>
</tbody>
</table>

References

