Chapter 3

Development of Artificial Neural Network for Geophysical Parameter Retrieval
Chapter 3

Development of Artificial Neural Network for Geophysical Parameter Retrieval

3.1 Introduction

This chapter deals with the theory of Artificial Neural Networks (ANNs) specifically the Back Propagation Neural Network (BPNN), an overview of neural network techniques applied to microwave radiometers and related retrieval algorithm.

3.2 Artificial Neural Network

ANNs exploit an analogy to the human brain. The idea behind ANN was to transfer the idea of parallel distributed processing, as found in the brain, to the computer in order to take advantage of the processing features of the brain. ANNs have been studied almost from the beginning of the computer era. ANNs are mostly referred to as Neural Networks (NNs). The field of artificial neural network has been made tremendous progress in the past 20 years in terms of theory, algorithms and applications. Notably, the majority of real world neural network applications have involved
3.2 Artificial Neural Network

the solution of difficult statistical signal processing problems. Compared to the con-
ventional signal processing algorithms that are mainly based on linear models, ANNs
offer attractive training algorithms.

The availability of such powerful modeling tools has motivated numerous research
efforts to explore the new signal processing applications. The nonlinear nature, ability
of learning from their environment in supervised and/or unsupervised ways, as well as
the universal approximation property make them highly suitable for solving difficult
signal processing problems, like the retrieval and time series prediction. From the
retrieval respective, it is imperative to develop a proper understanding of basic neural
network structure and its effectiveness in retrieval algorithms and applications. A
challenge in surveying the field of neural network paradigms is to distinguish those
neural network structures that have been successfully applied to solve real world
problems. In addition, it is also important to assess the impact of neural networks on
the performances, robustness of the systems and develop methodologies for integrating
neural network with other retrieval algorithms.

Estimating high quality geophysical parameters (information about physical, chem­
ical, and biological properties of the oceans, atmosphere, and land surface) from re­
 mote (satellite, aircraft, etc.) measurements is very important problem in geosciences
such as meteorology, oceanography, climatology and environmental modeling. The
quality of geophysical parameters derived from these measurements varies signifi­
cantly on the strength and uniqueness of the signal from the geophysical processes
and mathematical methods applied to extract these parameters, i.e. to solve forward
and inverse remote sensing problems.

The development of artificial neural networks started from the efforts to simulate
biological nervous systems by combining many simple computing elements neurons
3.2 Artificial Neural Network

into a highly interconnected system. A computer can be used to simulate a biological neural network. This computer simulated neural network is called an artificial neural network. A biological neuron cell, as shown in figure (3.1a) is the basic building block of a human brain. Figure (3.1b) is a computer model of neuron

(a) An human brain-neuron

(b) A artificial neuron

Figure 3.1: Neurons

A large NN might have hundreds or thousands of processing units whereas a mammalian brain has billions of neurons with a corresponding increase in magnitude of their overall interaction and emergent behavior. In practical terms, artificial neural networks are essentially very simple computer programs that can automatically find non-linear relationships/patterns in data without any pre-defined model form or domain knowledge. They consist of an often large number of neurons, i.e., simple linear or nonlinear computing elements, interconnected in often complex ways and often organized into layers. Artificial neural networks are mainly used as:

- models of biological nervous systems and intelligence
- real-time adaptive signal processors or controllers implemented in hardware for applications such as robots
3.2 Artificial Neural Network

- data analytic methods

The main feature of neural network is that they can learn the internal characteristic of a system by analyzing the datasets. The network then can reproduce or even predict new output of the system. These networks are good when the data is large, noisy, and has unknown relationships. The learning in a brain is based on synaptic modification of the strength of the connection between the neurons. So neural network resembles brain in two ways, firstly, the knowledge acquired through the learning process and secondly, the strength of connections between neurons, termed as synaptic weight used to store the knowledge.

3.2.1 Application of Neural Network

In general the main application of NN is data analysis. A NN requires dataset to be trained and provides solutions with better accuracies by learning the internal characteristic of the dataset. NN are mainly used in following applications, 1) Classification, 2) Forecasting and 3) Modeling and others.

Classification: These applications categorize various characteristics of the dataset. Some of the various applications are pattern recognition/pattern classification, speech recognition, natural language processing, expert system, data-mining application, image processing, and Remote Sensing.

Forecasting: These applications identify the time dependent behavior of the dataset to be used for predicting a certain variable in future, like function fitting and regression, function approximation, forecasting, and scheduling.

Modeling: Modeling application focuses on the modeling on certain intelligent processes like emotions, dialog, robot-control etc.
3.2 Artificial Neural Network

3.2.2 Advantages and Limitations of Neural Network

Like others techniques, NN also has certain advantages and limitations as mentioned below.

**Advantages:** NN is in general effective for dataset with missing point and noise, does not require prior information of approximate model or algorithms, suitable for systems with non-linear behavior, applicable to large dataset with numerous variables or parameters and has parallel distributed processing capability with multiple inputs and outputs. There are many practical advantages like computational speed, accuracy and robustness. In NN, all continuous bounded functions can be learned and the shape and location of the bounded functions can be adjusted. NNs are capable of self learning and generalization.

**Limitations:** NNs are computationally expensive during training phase, but are efficient in recall phase. Despite many advantages NN is considered as a "black box" because of the inability to explain comprehensively how a trained neural network reaches its output. NN does not provide information about the nature of the relationship between predictor and target variables, except a predicted value with some statistics about goodness of fit. Recently Zwaag et al. (2002) described domain-specific NN analysis methods that utilize domain-specific base functions which are easy to interpret by the user and can even be used to optimize NN systems. This NN analysis uses two-dimensional vectors applied to some well-known image filters, enabling comparison of conventional edge detectors known from literature and the neural network edge detectors. NN sometimes also suffer with the problem of over/under training which can be solved experimentally.
3.2 Artificial Neural Network

3.2.3 Building Blocks of Neural Network

Every neural network consists of a number of a building block. In general the network consists of neurons and connections. Neurons are organized in layers. The number of neurons, number of connections, type of connections, number and type of layers are all dependants on the classification of NN. Figure (3.2) shows one example of a possible neural network structure.

Neuron

A neuron (also called single perceptron) is the smallest unit of NN. Figure (3.3) shows the detail structure of a neuron. The neuron is a simple model of a real brain neuron. The neuron consists of various parts like input and output connections, synaptic weight, summing junction, activation function and a threshold value. The threshold value can be modeled as an offset input.

A simple perceptron computes a linear combination of the inputs (possibly an intercept or bias term) called the net input. Then an activation function is applied to the net input to produce the output. An activation function, maps any real input...
3.2 Artificial Neural Network

into a usually bounded range, often 0 to 1 or -1 to 1

\[ y_k = \varphi(u_k) \]

where

\[ x_0 = +1 \]
\[ w_{k0} = b_k \]

Figure 3.3 Single neuron

**Activation function**

The activation or transfer function decides the manner in which weighted inputs are connected to the output of a neuron. The activation function is necessary to introduce the non-linearity in the network. This non-linearity makes it possible to learn non-linear functions. For hidden layers in Multi Layer Perceptrons (MLP), usually sigmoidal functions, are preferred yielding best results in most cases. These functions are easier to change than the threshold functions because threshold functions usually do not change the output when the weights changes very little. A perceptron can have one or more outputs. Each output has a separate set of weights. Usually the same activation function is used for each output, although it is possible to use different activation functions. Perceptrons are most often trained by least squares, i.e., by attempting to minimize \( \sum \sum e_k^2 \), where the summation is over all outputs and
over the training dataset. A perceptron with a linear activation function is thus a linear regression model. A perceptron with a threshold activation function is a linear discriminant function (Hand, 1981, McLachlan, 1992, Weiss and Kulikowski, 1991), and a perceptron with a sigmoid activation function is thus a non-linear regression model.

Layers

Neurons are organized in layers like, one input layer, one or more hidden layers and one output layer. The input layer consists of neurons serving as input to the network. The neurons in the input layer transform the input data into information to be processed by NN. The hidden layers consist of neurons and are placed between the input and output layers. One or more layers can be designed with varying number of neurons in each layer. The hidden layers transform the information from input layer to the subsequent hidden layer(s) or to the output layer and the output layer consists of neurons, which transform the information from the last hidden layer to the output of the network.

Connections

Connection between various layers in a NN can be of four types. One lateral connections, connects neurons inside a layer, recurrent connections, connects a neuron to itself, and feed-forward connection, is unidirectional connections between neurons of consecutive layers.

NNs can be modeled into three broad categories, namely, feed-forward, feed-back, and cellular, which are further classified based on 1) pattern of connections between the neurons (also called its architecture or model), 2) activation function used in the
3.2 Artificial Neural Network

neurons, and 3) learning algorithm (criteria for determining weights)

All NN paradigms involve a training phase and a testing phase. In the training (learning) phase (usually offline) the NN is trained until it has learned its tasks while the testing (recall) phase is used to realize the task. Some NN paradigms are named after their proposer such as Hopfield, Kohonen, etc. Most NNs are named after their learning algorithm such as Backpropagation, Competitive learning, Counter propagation, conjugate gradient, ART, etc. and some are named after their model such as BAM basically a particular NN. A NN classification is shown in Table (3.1).

Table 3.1: Neural network model classification

<table>
<thead>
<tr>
<th>NN Models</th>
<th>Feed forward</th>
<th>Feed back</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervised</td>
<td>Least Mean Square</td>
<td>Recursive backpropagation</td>
</tr>
<tr>
<td></td>
<td>Backpropagation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reinforcement learning</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cascaded Correlation</td>
<td></td>
</tr>
<tr>
<td>Unsupervised</td>
<td>Self-organizing maps (KNN)</td>
<td>Adaptive resonance theory</td>
</tr>
<tr>
<td></td>
<td>Competitive learning</td>
<td>Fuzzy ART</td>
</tr>
<tr>
<td></td>
<td>Counter propagation</td>
<td>Boltzmann learning</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Hopfield network</td>
</tr>
</tbody>
</table>

3.2.4 Learning Law

There are a variety of commonly used learning laws. These laws are mathematical algorithms used to update the connection weights. The process of finding the best set of weights for the NN is referred to as training or learning. The approach used most,
to estimate the weights is backpropagation. Each time the network cycles through the training data, it produces a predicted value for the target variable. This value is compared to the actual value of the target variable and an error is computed for each observation. The errors are fed-back ("feed back") through the network and new weights are computed to reduce the overall error. Despite the neural network terminology, the training process is actually a statistical optimization procedure. Typically, the procedure minimizes the sum of squared residuals. The human brain basically learns from experience. NNs are sometimes called machine learning algorithms, because changing of its connection weights (training) causes the network to learn the solution to a specific problem. The strength of connection between the neurons is stored as a weight for that specific connection. The system learns new knowledge by adjusting these connection weights. The learning ability of a neural network is determined by its architecture and by the algorithmic method chosen for training.

Two types of learning prevailed in NNs, supervised an unsupervised learning.

**Supervised learning:** it is performed using the teacher (target) signals. Difference between the NN output and the target treated as error signal, is then minimized through continuous adaptation of the weights to solve the problem through a learning algorithm. When the error found with in the acceptable limits, the NN is assumed to have learned the task and the training process is stopped. In the thesis, for retrieval purpose the dataset used to train the NN is the brightness temperature simulated through radiative transfer model.

**Unsupervised learning:** In this case, the neurons must find a way to organize themselves without the help of target signal. Most of these learning laws are a sort of variation of the best known and oldest learning law, the Hebb's Rule. Some of these learning laws are Hebb's Rule, Hopfield Law, Delta Rule, and Kohonen's.
3.3 The Back-Propagation Neural Network

The back-propagation algorithm is the learning strategy for NNs (Rumelhart et al., 1986). The back-propagation learning law is very popular for inversion related problems, and are often referred to as the Back-Propagation Neural Network (BPNN). It has also been used in this research. In BPNN, the weights are modified to reduce the difference between the target value and the output of NN. This rule changes the connection weights in a manner that minimizes the mean squared error of the network. The error is back propagated into previous layers one layer at a time. Thus back-propagation algorithm cycles through two distinct passes, a forward pass followed by backward pass through these layers of the network. The process of back propagating the network errors continues until the first layer is reached. The algorithm alternates between these passes several times for a given input/output vector and modifies the NN weights till it reaches the minimum desired error, i.e., the minimum error limit. This process consumed several iterations (this is also termed as local iteration as it is used in minimizing of a single vector). This iteration is part of the global iteration counter set as maximum number of iteration as the input parameters of NN model. Iteration counter continues as it scans the training data one by one till the NN scans the entire training data once, at this point NN is said to complete one epoch. NN iterates through several epochs till it reaches the minimum desire RMS error for the parameters under investigation.

Data can be fed to NN for training in two ways, pattern training mode and batch training mode. In the pattern training mode each input/output vector data point is fed to NN and error is minimized iteratively through back propagation for this vector (weight is updated in each iteration) and similarly the entire dataset is processed. In batch training mode the weight is updated only after all the input/output vectors
3.3 The Back-Propagation Neural Network

are processed after each epoch. In both the training mode typically the whole set of training data is scanned several times before the networks learns to make good learning. Pattern training algorithm is faster and more effective than the batch procedure especially for large database (Cichoki and Unbehaue, 1993).

3.3.1 Learning Algorithms for Single Perceptron

To explain the backpropagation algorithm in its basic form, let us first consider the learning of a single neuron as shown in figure (3.4). The hyperbolic tangent function as given by equation (3.3.1) is used as an activation function.

\[
y_j = \psi(u_j) = \tan(u_j) = \frac{1 - e^{-2u_j}}{1 + e^{-2u_j}} \tag{3.3.1}
\]

where \( u_j = \sum_{i=1}^{n} W_{ji} x_i + \Theta_j \) with \( W_{j0} = \Theta_j \) and \( x_0 = +1 \).

and

\[
\xi_j = \frac{1}{2} (d_j - y_j)^2 = \frac{1}{2} \xi_j^2 \tag{3.3.2}
\]

Figure 3.4: Single perceptron learning
3.3 The Back-Propagation Neural Network

Where \(d_j\) is target value and \(y_j\) is NN output. The aim of learning is to minimize the square error (also known as cost function) by modifying the weight \(W\), and need to determine how to increase or decrease the weights to minimize the local error function \(\xi_j\). This can be achieved using a steepest descent gradient rule

\[
\frac{dW_{ji}}{dt} = -\eta \frac{\partial \xi_j}{\partial W_{ji}} \tag{3.3.3}
\]

where \(\eta\) is a positive learning parameter determining the speed of convergence, \(\eta\) is an important tuning parameter that is chosen by trial and error by repeated runs on the training data. Typical values for \(\eta\) are in the range 0.1 to 0.9. Lower values give slow but steady learning while higher values give erratic learning and may lead to an unstable network. Applying chain rule of differentiation to the right hand side of eqn (3.3.3) yields the following expression

\[
\eta \frac{d\psi(u_j)}{du_j} \cdot \frac{\partial e_j}{\partial u_j} = \eta \delta_j \cdot y_j
\]

\[
W_{ji} = \frac{\partial e_j}{\partial W_{ji}} = \frac{\partial e_j}{\partial \psi(u_j)} \cdot \frac{\partial \psi(u_j)}{\partial u_j} = \eta \delta_j \cdot y_j\tag{3.3.4}
\]

Where \(\delta_j\), called the learning signal or local error, is expressed as

\[
\delta_j = \epsilon_j \psi'(u_j) = -\frac{\partial \xi_j}{\partial u_j}\tag{3.3.5}
\]

for the sigmoid activation function to be a hyperbolic tangent function given by equation (3.3.1), then the derivation \(\psi'(u_j)\) is given by

50
3.3 The Back-Propagation Neural Network

\[
\psi'(u_j) = \frac{d\psi(u_j)}{du_j} = [1 - (\tanh u_j)^2] = (1 - y_j^2)
\]

Equation (3.34) can be written in the form

\[
\frac{dW_j}{dt} = \eta e_j(1 - y_j^2)x_i = \eta_j e_j(1 - y_j^2)x_i
\]

with \( \eta_j > 0 \) that weight update stabilized if \( y_j \) approaches -1 or +1 since the derivative \( \partial y_j / \partial u_j \), equals to \( (1 - y_j^2) \), reaches its maximum for \( y_j = 0 \) and its minimal for \( \pm 1 \).

The weights are usually changed incrementally and the neuron gradually converges to a set of weight, which solves the specific problem. Change in weight \( W_j \) can be determined by

\[
\Delta W_{ji}(k) = W_{ji}((k + 1)\tau) - W_{ji}(k\tau) = \eta_j \delta_j x_j
\]

The weights are usually changed incrementally and the neuron gradually converges to a set of weight, yielding the solution to a specific problem. Change in weight \( W_j \) can be determined by

\[
W_{ji}(k + 1) = W_{ji}(k) = \Delta W_{ji}(k)
\]

The above describes the approach of adaptive learning of the weight which can be extended to multi-layer perceptron (MLP) as shown in figure (3.5).
3.3.2 Function Minimization Evaluation

In pattern training mode the RMS error of the entire dataset as compared to the target values is monitored and the training process is iterated till the desired RMS error is achieved. The RMS error of the dataset is defined by

\[ RMS_{\text{Error}} = \sqrt{\frac{\sum_{i=1}^{n} (d_i - t_i)^2}{n}} \]  

(3.3.10)

Where, \( n \) is the number of training cases. Along with the RMS error, bias between
3.4 Development of NN Algorithm for Geophysical Parameter Retrieval

the target and predicted values given by eqn. (3.3.13) is also monitored and the training process is resumed with bias correction incorporated at appropriate stage.

\[
Bias = \frac{1}{n} \sum_{i=1}^{n} (d_i - \hat{y}_i)
\]  

(3.3.11)

3.4 Development of NN Algorithm for Geophysical Parameter Retrieval

As discussed in Chapter 2, the satellite observations are related to various geophysical parameters through the process of radiative transfer hence a relationship exists between geophysical parameter and satellite observed radiation which is required to be determined using various inversion techniques. Among various inversion techniques, the ANN technique has been chosen for developing retrieval algorithm for geophysical parameters. For selecting the appropriate retrieval model it is necessary to have information about the dominant dependency of satellite observations on various geophysical parameters. This is explained by a specific example of sensitivity of SMMR channels on various geophysical parameters as given in figure (3.6) (Wilheit, 1979a).

As seen in figure (3.6), the sensitivities of sea surface salinity and temperature are maximum around 1 and 6 GHz, respectively. While sea surface wind has maximum sensitivity beyond 10 GHz. The sensitivity of atmospheric water vapour is maximum around 22 GHz due to water absorption line at 22.235 GHz. The sensitivity of cloud liquid water increases with frequency. Moreover, the radiation observed by the radiometer at a given frequency/channel is also affected by parameters other than the parameter having maximum sensitivity implying that multiple channels should be used for better retrieval.
3.4 Development of NN Algorithm for Geophysical Parameter Retrieval

Apart from this the satellite data is also available at grids with varying spatial resolutions which also contains different channels for different resolutions, for example, for IRS-P4 MSMR for 150 km resolution grid all the channels (6, 10, 18 and 21 GHz) are available while for 50 km resolution grid only 18 and 21 GHz channels are available which restrict the retrieval of different geophysical parameters under different grid resolutions like only water vapour and cloud liquid water is possible for 50 km grid while sea surface temperature, sea surface wind speed, water vapour and cloud liquid water are possible for 150 km grid.

The sensitivity analysis suggests that different combinations of channels are suitable for retrieving different geophysical parameters. Hence retrieval models for various parameters may have different channels as predictors. In view of availability of different channels and varying sensitivity suitable NN models can be designed. Due to

![Diagram of sensitivity analysis](image)

Figure 3.6: Sensitivity of $T_B$ to geophysical parameters, the arrow indicates the SMMR frequencies, Wilheit (1979a)
the special capability of NN of mapping multiple inputs to single or multiple outputs, it is also possible to develop single or multi-parameter retrieval models. As reported by (Krasnopolsky et al., 1999), single-parameter algorithms may have additional systematic bias and unknown component of RMS errors e.g. SSM/I algorithms for wind speed (Goodberlet et al., 1989), for water vapor (Alishouse et al., 1990), and for cloud liquid water (Petty, 1993). The obvious way to improve single-parameter retrievals is to include the other parameters in the retrieval process. A typical example of NN models using single and multi-parameter developed for IRS-P4 MSRR is depicted in Table (3.2). These NN based retrieval models have been classified according to the single or multi-parameter retrievals termed as class-I and class-II models respectively. Figures (3.7, 3.8) depict various models developed for IRS-P4 MSRR.

Table 3.2. Different NN Models

<table>
<thead>
<tr>
<th>Models</th>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class-I (Single Output)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4Tₜ - 1GP</td>
<td>Tₜ₁₈V, Tₜ₁₈H, Tₜ₂₁V, Tₜ₂₁H</td>
<td>WVC / CLW</td>
</tr>
<tr>
<td>6Tₜ - 1GP</td>
<td>Tᵢ₀₅V, Tᵢ₀₅H, Tᵢ₁₀V, Tᵢ₁₀H, Tᵢ₁₈V, Tᵢ₁₈H</td>
<td>SSW / SST</td>
</tr>
<tr>
<td>8Tₜ - 1GP</td>
<td>Tᵢ₀₅V, Tᵢ₀₅H, Tᵢ₁₀V, Tᵢ₁₀H, Tᵢ₁₈V, Tᵢ₁₈H</td>
<td>WVC / CLW / SSW / SST</td>
</tr>
<tr>
<td>Class-II (Simultaneous Output)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4Tₜ - 2GP</td>
<td>Tᵢ₁₈V, Tᵢ₁₈H, Tᵢ₂₁V, Tᵢ₂₁H</td>
<td>WVC, CLW</td>
</tr>
<tr>
<td>6Tₜ - 2GP</td>
<td>Tᵢ₀₅V, Tᵢ₀₅H, Tᵢ₁₀V, Tᵢ₁₀H, Tᵢ₁₈V, Tᵢ₁₈H</td>
<td>SSW, SST</td>
</tr>
<tr>
<td>8Tₜ - 2GP</td>
<td>Tᵢ₀₅V, Tᵢ₀₅H, Tᵢ₁₀V, Tᵢ₁₀H, Tᵢ₁₈V, Tᵢ₁₈H</td>
<td>WVC, CLW / SSW / SST</td>
</tr>
<tr>
<td>8Tₜ - 4GP</td>
<td>Tᵢ₀₅V, Tᵢ₀₅H, Tᵢ₁₀V, Tᵢ₁₀H, Tᵢ₁₈V, Tᵢ₁₈H</td>
<td>WVC, CLW, SSW, SST</td>
</tr>
</tbody>
</table>

In general, the radiance measurement involves responses of detector system and electronics of sensor system as well as scanning systems (involving integration times). Therefore, the observations are laced with "measurement errors". These measurement
3.4 Development of NN Algorithm for Geophysical Parameter Retrieval

Figure 3.7: Class-I neural network model configurations

(a) (4TB-1GP)

(b) (8TB-1GP)

Figure 3.7: Class-I neural network model configurations
3.4 Development of NN Algorithm for Geophysical Parameter Retrieval

Figure 3.8: Class-II neural network model configurations

(a) \(4T_B-2GP\)

(b) \(8T_B-4GP\)

Figure 3.8: Class-II neural network model configurations
3.4 Development of NN Algorithm for Geophysical Parameter Retrieval

errors specify the accuracy of possible retrievals as well as utility of particular spectral channel(s) for a particular geophysical parameter. Hence, all retrieval schemes are subjected to noise ($\Delta T_B$) considerations statistically. The solution to optimization problem performed better when sensor-specific anticipated noise is introduced. For each sensors channels, Gaussian noise with certain standard deviation in their respective channels and with zero mean is added by means of a random noise generator. While developing retrieval model, such noise are incorporated in their respective channels and this database is actually used in model development. This added noise will ensure the robustness of the retrieval model to perform in the real world scenario.

For MSMR NN model development the simulated $T_B$ have been individually perturbed with Gaussian noise for different channels with certain standard deviation. The noise figures are preliminary in nature, which has been evaluated from the limited MSMR data based on simulated data using the minimized approach used in MSMR operational algorithm (Gohil et al., 2000).

Radiative transfer-based simulations have been used for NN modeling, and statistical analysis for selecting the final NN configuration and the results of this analysis are presented in Chapter 4.