Chapter 6

Neural Networks Based Selection of Echo Features

6.1 INTRODUCTION

An Artificial Neural Network (ANN) is a nonlinear information processing system especially useful for mapping input vectors to specific outputs, where no hard and fast rules are applicable. An ANN uses mathematical algorithms to learn the relationships and the patterns in a given dataset.

ANN is an analogy of a biological neuron (Hertz et al., 1991). The schematic diagram of a biological neuron is shown in Fig. 6.1. A human brain is composed of about $10^{11}$ neurons (nerve cells). Tree like networks of nerve fiber is called dendrites and these are connected to a cell body, which contains nucleus. A single long fiber extending from the cell body is called axon, which eventually branches into strands and sub-strands. The ends of these branches are called synaptic junction or synapses, which transmit signals to the other neurons. The receiving ends of these junctions on other cells may be located on the dendrites as well as on the cell bodies. A few thousand synapses exist on the axon of a typical neuron. The transmission of a signal from one cell to the other through synapses is a complex chemical process. Specific chemicals are released from the sending side of the junction to raise or lower the electrical potential
inside the body of a receiving cell. When this potential reaches a threshold value, a pulse, called action potential, with certain strength and duration is fired through the axon. The pulse moves from one cell to the other through these synapses. After firing, a cell waits for a specific time, called the refractory period, before the next firing.

The aim of an artificial neural network is to imitate neurons in a brain and its ability to adapt to the current situation and the changing circumstances. The performance of a neural network depends heavily on the ability to learn from the past events and apply this information to a future situation. In general, a neuron receives inputs from the outside, performs non-linear operations on them, and then outputs the final results. The fundamental processing element of a neural network is called neuron.

In ANN, several neurons (also called nodes) are interconnected according to some topology to perform a specific task. The output of each neuron may be given to several neurons. The amount of output from one neuron received by another neuron depends on the strength of the connection between the neurons and it is called weight (or synaptic weight) of the connection links. At any instant of time, each neuron has a unique activation value and a unique output value. In a typical operation, each neuron of an ANN receives inputs from other interconnected neurons and/or from the external
source. A weighted sum of the inputs and a specific activation value determine the actual output of an ANN. Therefore, an artificial network consists of architecture (i.e., the topology) of a network, training or learning algorithm (of the weights between neurons), and activation function (Beale and Jackson, 1990).

Fundamentals of artificial neural networks are covered in several books (Beale and Jackson, 1990; Hertz et al., 1991; Yegnanarayana, 2001; Sivanandam et al., 2006). However, the basics of neural networks and in particular backpropagation networks are presented in this chapter. Later, an analysis on the relative importance of different subsets of echo features for seafloor classification using multilayer perceptron networks (with backpropagation training) is presented.

6.1.1 ANN Terminologies

6.1.1.1 Weight

It is mentioned that the strength associated with the interconnection between two neurons is termed as weight of that connection. Weight in a neural network is the information used by the network to solve a specific problem. The inputs to a neuron may come from the outputs of other neurons or from external sources. The amount of output of one neuron received by another neuron depends on the strength of the interconnection (or weights) between the neurons.

6.1.1.2 Activation Function

The activation function (also called the transfer function) determines the actual output from a neuron. The activation function may be linear as well as non-linear. Each
input signal (let’s say $x_i$) is multiplied by a weight (let’s say $w_i$) and the product is summed. The summation of the products is called $NET$

$$NET = \sum_i x_i w_i$$ \hspace{1cm} (6.1)

An activation function is applied to modify the sum of the weighted input signal (i.e., $NET$) to produce the output signal. The activation function is differentiable at every point. This function does not allow the output to exceed a very low and a very high limit regardless of the value of $NET$, and hence it is also called a squashing function. If the function is linear, then the output is equal to $NET$. A non-linear activation function is generally used to map a non-linear process. In general, a logistic or a squashing function, called a sigmoid function (S-shaped function), is used because of its self-limiting nature (Beale and Jackson, 1991; Hertz et al., 1991). The sigmoid function is given by

$$y = F(NET) = \frac{1}{1+e^{-NET}}$$ \hspace{1cm} (6.2)

This is also called a log sigmoid (or logsig) function. The logsig function compresses the inputs into a range within 0 to 1 (Fig. 6.2). Basically, the activation function acts as a nonlinear automatic gain control for an artificial neuron. The magnitude of the gain is the slope of the curve (of the activation function) at a specific excitation level. For a small value of $NET$ (near to zero), the gain is very high. This central high gain region of a logistic function is useful to solve the problem of processing small signals, whereas the extreme positive and negative regions with low gain are appropriate for large excitations.

Another activation function is the bipolar sigmoidal function. This function is called a hyperbolic tangent function (Hertz et al., 1991) and is given by

$$y = F(NET) = \tanh(NET) = \frac{e^{NET} - e^{-NET}}{e^{NET} + e^{-NET}}$$ \hspace{1cm} (6.3)
The hyperbolic tangent function is symmetrical about the origin within a range +1 to -1. When the value of $NET$ is zero, the output is also zero. This is also called tan sigmoid (or *tansig*) function. Few more transfer functions are also available such as *purelin*, *hardlim*, *poslin*. *Purelin* is a linear function, where inputs and outputs are linearly related. *Hardlim* function has only two outputs either 0 or 1. The output of *poslin* function assumes any positive value, linearly scaled without any restriction of range. The transfer function *poslin* returns the output $n$, if $n$ (an integer) is greater than zero and 0 if $n$ is less than or equal to zero. The shapes of four activation functions (*tansig*, *purelin*, *logsig*, and *hardlim*) are shown in Fig. 6.2.

**6.1.1.3 Bias**

Bias is a weight, which is always initialized to a value of 1. It is often advantageous to have a bias weight for rapid convergence of a training process. Bias
weight allows a neuron to have an output even if the input is zero. The main purpose of
a bias is to shift the origin of an activation function. These weights are trainable just as
the other weights. If a bias \((b)\) is present, then the \(\text{NET}\) is calculated as (Sivanandam et
al., 2006),
\[
\text{NET} = b + \sum_i x_i w_i
\] (6.4)

### 6.1.1.4 Threshold

The threshold is a limiting value of a neuron to produce an output. The weighted
sum of inputs must reach or exceeds the threshold value (Hertz et al., 1991) for a neuron
to fire (i.e., to get an output). The binary step function is an example of the threshold
function. In the presence of threshold \((\theta_{th})\), the equation (6.4) is written as,
\[
\text{NET} = \left( b + \sum_i x_i w_i \right) - \theta_{th}
\] (6.5)

### 6.1.1.5 Training

Training is a process to modify the weights of interconnection between various
layers of a network with an objective to achieve expected output (Sivanandam et al.,
2006). The process that takes place inside a network during training is called learning.
There are three types of training such as supervised training, unsupervised training, and
reinforcement training. Supervised training of a network is the process of learning when
the expected target outputs (response vectors) are available. Examples of supervised
training algorithms are: Hebb net, backpropagation net etc. If the expected target output
is not available, the training method adopted is termed as unsupervised training. In the
unsupervised training, the weights of the network are adjusted in such a way that the
similar input vectors are assigned to the same output unit. The reinforcement training is a type of supervised training. The desired output vector is not available in this case, but the condition whether the output is a success or failure is indicated. The network uses this information to improve its performance to learn the input-output mapping through a trial and error process. The supervised backpropagation learning method is discussed in this chapter. Other methods, called Kohonen’s competitive unsupervised and supervised learning are discussed in Chapter 7.

6.1.2 Fundamental Model of Artificial Neural Network

McCulloch and Pitts (1943) formulated a synthetic neuron model, based on the concept of a simplified biological model. The input values are connected to neurons either by excitatory (positive) or inhibitory (negative) weights (Fig. 6.3). A neuron fires if the net input to the neuron is greater than a threshold value. Any number of inputs can be added to a neuron. In Fig. 6.3, the inputs $x_i$, $i = 1, \ldots, n$ are connected to neurons by excitatory weights $w_i$, and the inputs $x_j$, $j = n+1, \ldots, n+m$ are connected to neurons by inhibitory weights $w_j$. The output signal of McCulloch and Pitts model is expressed by the following equations:

\[
y = F(NET),
\]

where

\[
NET = (\sum_i w_i x_i + \sum_j w_j x_j) - \theta_{th}
\]

The function $F$ is called the activation function, $\theta_{th}$ is the threshold, and $NET$ is the total net input signal received by the neuron (without bias term).
The activation function in McCulloch-Pitts is given by (Hertz et al., 1991),

\[ F(NET) = \begin{cases} 
1, & \text{if } NET \geq \theta_{th} \\
0, & \text{if } NET < \theta_{th} 
\end{cases} \]  \hspace{1cm} (6.8)

### 6.1.3 Perceptrons

Layered feed-forward network is called a perceptron. Rosenblatt (1962) and Minsky and Papert (1969) developed this network. There were three layers in the original perceptron e.g., sensory unit, association unit, and response unit (Fig. 6.4a). The sensory and association units have binary activations. The response unit uses activations of +1, 0, or -1. All the units have their own weights. The association unit performs the predetermined mathematical operations on its inputs. The difference between the McCulloch-Pitts model and the perceptron model is that the learning function (adjustment of weights) is introduced in the perceptron (Fig. 6.4b). The desired or target output \( T \) is compared with the actual output \( y \), and the error \( \delta \) is calculated to adjust the weights. The output signal is given by \( y = F(NET) \), where \( NET = (\sum w_i x_i + b) - \theta_{th} \). The error term is calculated as, \( \delta = (T - y) \). If the error associated with the input vector \( x_i \) is \( \delta x_i \), then the change in weight \( \Delta w_i \) is expressed as, \( \Delta w_i = \eta \delta x_i \), where \( \eta \) is called the learning rate parameter (Yegnanarayana, 2001).
A single layer perceptron is the simplest form of a neural network. This type of network model is generally used for the classification of patterns that are linearly separable. There exists another class called MultiLayer Perceptron (MLP). This type of network consists of a set of sensory units with an input layer and one or more hidden layers. These hidden layers are useful for performing complicated tasks, but at the cost of a lengthy learning process. Feed forward network is an example of MLP network.

### 6.1.4 Network Architectures

The arrangement of neurons into layers and the pattern of connections among the neurons in various layers are termed as neural network architecture (Beale and Jackson, 1990). Schematic diagrams of a simple network and a multilayer network are shown in Fig. 6.5a and Fig. 6.5b respectively. There is no maximum limit to the number
of layers or the number of neurons in a layer. However, computational requirement increases with the increase in the number of neurons (and weights). There are various types of network architectures such as feed forward net, feedback net, competitive net, and recurrent net.

![Schematic diagram of (a) a simple network (b) 3 layers network architectures](image)

**Fig. 6.5** Schematic diagram of (a) a simple network (b) 3 layers network architectures

In a feed forward network, signal propagates in the forward direction from input layer to output layer i.e., in this network signal travels in only one direction from the input to the output. There are no feedback loops i.e., the output of any layer does not affect the same layer. The feed forward network can be a simple network, where inputs are directly connected to the outputs through only one layer of weighted interconnections. It may also have multiple layers with one or more hidden layers between the input and the output layers. Multilayer network (as shown in Fig. 6.5b) is advantageous over a single layer network to solve complicated problems. Backpropagation network is one of the most important types of feed forward network.

In a feedback network, the signals can travel in both the direction through the network. These networks are dynamic i.e., the state changes continuously till they reach an equilibrium position. A general form of a feedback network consists of a set of processing units, where the output of each unit is fed as input to all the other units
including the same unit. A feedback network does not have any specific structure (Yegnanarayana, 2001). Hopfield network is an example of a feedback network.

A competitive network (Fig. 6.6) is similar to a feed forward network. The difference is that there are connections between the output neurons in a competitive network. Due to these interconnections, the output nodes tend to compete each other to represent an input pattern. The output nodes are connected to each other (or connected to the neighborhood nodes only).

In a fully recurrent network, all the nodes are connected to the rest of the nodes and every node acts as an input as well as an output (Fig. 6.7). For each input pattern, the weights of all the units are modified. The advantage is that if a degraded version of one of the patterns is presented as input, the network tries to reconstruct the true pattern. The examples of recurrent net are the simple recurrent network and the Jordan network.

![Fig. 6.6 Schematic diagram of a competitive network](image)

![Fig. 6.7 Schematic diagram of a fully recurrent network](image)
6.2 BACKPROPAGATION NETWORK

Backpropagation network is a type of multilayer feed forward network. Rumelhart et al. (1986) first introduced the backpropagation network. Backpropagation network is an efficient supervised learning method to capture the inherent characteristics of a given set of input-output pairs. As the backpropagation method is based on an error-correction learning rule, it is also known as error backpropagation algorithm. It consists of two processes: a forward process and a backward process (Fig. 6.8). In the forward process, a set of outputs is produced through the applications of activation functions to the inputs of a network. During the forward process, fixed synaptic weights are used at different layers of the network. The error signal (deviation of the actual output from the desired response) is propagated backward through the network. The synaptic weights are adjusted (when a signal propagates in the backward direction) based on some form of the error-correction rule (Hertz et al., 1991).

Fig. 6.8 Schematic diagram of a single hidden layer backpropagation network. The solid lines indicate forward propagation of signals and the dashed lines indicate backward propagation of errors ($\delta_i$).

Backpropagation network utilizes the feed forward network with differentiable activation function. The training of a backpropagation network is based on some
algorithm to minimize the total error of the network output. The error function is a measure of the deviation between the desired and the actual output of a network. The mean square error for the \( m^{th} \) training pattern is defined (Masters, 1993) as

\[
E_m = \frac{1}{N} \sum_{j=1}^{N} (T_{mj} - O_{mj})^2,
\]

(6.9)

where \( E_m \) is the error for the \( m^{th} \) training pattern; \( T_{mj} \) is the desired (correct) output value of the \( j^{th} \) output neuron; and \( O_{mj} \) is the actual output from the \( j^{th} \) output neuron; \( N \) is the total number of output neurons. Therefore, the total error \( E \) for all the training patterns is obtained from \( E = \sum_mE_m \). By squaring the absolute error (between the desired and the actual output in the equation (6.9)), it is ensured that the distant output from the desired value contributes more strongly in the summation of total error.

### 6.2.1 Backpropagation Training Algorithms

There are number of training algorithms, which can be used with a backpropagation network. The algorithms namely gradient descent method, variable learning rate backpropagation with momentum, Levenberg-Marquardt backpropagation, and resilient backpropagation methods are commonly used. These methods are introduced in the following sections.

#### 6.2.1.1 Gradient Descent Method

Gradient descent method, also called the steepest descent method, utilizes a negative gradient of an error function with respect to the weights for rapid reduction of
the error function. The gradient \( \Delta w_{ji} \) i.e., the change in the weight \( w_{ji} \) for the \( i \)th source neuron in a layer to the \( j \)th destination neuron in the next layer is expressed as,

\[
\Delta w_{ji} = - \frac{\partial E_m}{\partial w_{ji}},
\]

(6.10)

where \( E_m \) is the error for the \( m \)th input training vector. This can be expressed as,

\[
\Delta w_{ji} = \eta \delta_{mj} O_{mi}
\]

(6.11)

Here the constant of proportionality, \( \eta \) is known as the learning rate and it governs the distance moved in the direction of negative gradient at each step. \( O_{mi} \) is the output of the \( i \)th neuron for the \( M \)th training vector. \( \delta_{mj} \) is the error at the \( j \)th output neuron in a layer for the \( m \)th training vector and is expressed as,

\[
\delta_{mj} = (T_{mj} - O_{mj})O_{mj}(1-O_{mj}), \text{ For output neurons}
\]

(6.12)

\[
\delta_{mj} = O_{mj}(1-O_{mj})\sum_n \delta_{mn} w_{nj}, \text{ For hidden neurons}
\]

(6.13)

Here \( T_{mj} \) is the desired output value from the \( j \)th neuron; \( O_{mj} \) is the actual output of the \( j \)th neuron; and \( \delta_{mn} \) is the error signal at the \( n \)th neuron in the hidden layer. Finally the change in weights \( \Delta w_{ji}(p+1) \) at \((p+1)\)th iteration are adjusted by utilizing the change in weights \( \Delta w_{ji}(p) \) at \( p \)th iteration. This is expressed by the equation,

\[
\Delta w_{ji}(p+1) = \eta \delta_{mj} O_{mi} + \alpha_m \Delta w_{ji}(p),
\]

(6.14)

where \( \alpha_m \) is called momentum constant (lies within 0 and 1) and is used for rapid convergence. To speed up the convergence time, the variable learning rate is generally used. A larger learning rate is utilized during training when the neural network model is far from the desired target and a smaller learning rate is used when the neural model approaches towards the desired target.
6.2.1.2 Levenberg-Marquardt Algorithm

Levenberg-Marquardt algorithm minimizes the total error of a network by solving the equation,

\[
(J^T J + \mu_L I)\Delta w_{ji} = J^T E_m
\]  \hspace{1cm} (6.15)

where \( J \) is the Jacobian matrix containing first derivatives of the network errors with respect to the weights; \( J^T \) is the transpose of \( J \); \( I \) is the identity matrix; and \( \mu_L \) is the Levenberg's damping factor controlling the behavior of the algorithm. The new weights at \( (p+1) \) iteration are calculated as,

\[
w_{ji}(p+1) = w_{ji}(p) - \Delta w_{ji}(p) = w_{ji}(p) - (J^T J + \mu_L I)^{-1} J^T E_m
\]  \hspace{1cm} (6.16)

The term \( J^T J \) is called Hessian matrix and it is computationally very intensive.

The Levenberg-Marquardt algorithm is very sensitive to the initial weight vectors used for training a network. A network can only be solved with the Levenberg-Marquardt algorithm if the Hessian matrix is not singular.

6.2.1.3 Resilient Backpropagation Algorithm

Sigmoid transfer functions are generally used in the hidden layers of a multilayer network. If the inputs to these functions are large, the slopes of these functions approach to zero. This creates a problem with the gradient approach based algorithms for training a neural network with a sigmoid function. Therefore, a small gradient makes small changes during the adjustment of weights and biases, even if the weights and biases are far from their optimum values.
Resilient backpropagation algorithm eliminates these limitations of partial derivatives based algorithms. In this algorithm, only the sign of the derivative is used to determine the direction of the weights update; and thus the magnitude of the derivative does not play any role in the weights update. The magnitude of the change in weights is controlled by a separate update value. When the derivatives of two successive iterations have the same sign, the update value for each weight and the bias is increased by a separate value. This update is also called the increment in weight change. The update value is decreased (called the decrement in weight change) when the sign of the derivative changes from the previous iteration. If the derivative is zero, then the update value remains unchanged. If the change in weight continues in the same direction for several iterations, then the magnitude of the change in weight is increased. A function 'trainrp' is available in the Neural Network Toolbox of MATLAB 7.0 (2004), which uses resilient backpropagation algorithm for training a network.

One complete presentation of all the input vectors (i.e., pattern) available in a training set is called the epoch. When the total error of a network reaches or falls below a pre-defined stopping criterion (either based on the maximum number of epoch or the pre-defined acceptable error limit), the network is said to have converged. Several such epochs are necessary to train a network for solving a specific problem.

6.2.2 Performance of a Neural Network

The performance of any neural network is expressed by an error between the predictions of a network and the true value. The Mean Square Error (MSE) is one of the methods to measure this error. A test dataset is generally used for evaluating the performance of a trained network. During the testing, it is assumed that the true values
of outputs (the desired outputs) are known in advance. The MSE is obtained from the difference between the desired target output and the achieved output by a network (as given in the equation (6.9)).

6.3 MLP NETWORKS BASED FEATURES SELECTION

One of the most widely used neural networks in the classification of seafloor sediments is the multilayer perceptron (MLP) with backpropagation (Stewart et al., 1992; Alexandrou and Pantartzis, 1993; Stewart et al., 1994; Michalopoulou et al., 1995; Chakraborty, 2002; Chakraborty et al., 2003a). As mentioned earlier that most of the neural network based seafloor classification techniques use echo features as inputs, which are usually derived from the seafloor backscatter data. Selection of echo features as input variables to a neural network is an important criterion for achieving the higher success in the classification. Improper selection of the input features leads to difficulties in converging the training of a neural network (thus increasing the computational time). As such there is no general rule for deciding the best features for a given problem (Chakraborty, 2002). It is widely accepted that the selection of input features mainly application oriented and depends on the physical processes under study (Stewart et al., 1994).

In the present analysis, 7 echo features (as discussed in Chapter 5) namely: BS, SpSkew, SpKurt, SpWidth, TS, StatSkew, and HD are utilized. The relative importance of different subsets of echo features (among the 7 features) for the classification of seafloor sediments using MLP networks (trained with resilient backpropagation algorithm) is investigated. A set of features chosen out of the 7 echo features is termed as the feature subset in this work. An optimum subset of features with dominant characteristics is
decided on by analyzing the effects of different subsets of echo features on the performance of a given MLP network.

6.3.1 Pre-Processing of Input Data

Preparation of input data is one of the important aspects in neural network analysis. A uniform scaling is essential to equalize the importance of all the variables. The training algorithm uses the total error (from all the outputs) minimization scheme. If the output variables are unequally scaled, the variables with larger variability will be given importance, as these will dominate the total error.

To constrain the range of each input variable, the input data are often rescaled to a new uniform range of values. In this study, the input data are scaled in such a way that all the values lie within an interval -1 and +1 using the relationship (Neural Network Toolbox in MATLAB 7.0, 2004),

\[ X_s = 2 \left\{ \frac{(X - \text{Min}X)}{\text{Max}X - \text{Min}X} \right\} - 1 \]  

(6.17)

where, \( X \) is the input data, \( \text{Min}X \) is the minimum value of \( X \), \( \text{Max}X \) is the maximum value of \( X \), and \( X_s \) is the scaled output of \( X \). Moreover, scaling of data is carried out in such a way that the data used in training are proportional with that used for testing the network.

6.3.2 Methodology

Previous studies indicate that a feed-forward neural network with one hidden layer is sufficient to solve a majority of practical problems (Masters, 1993). Hence, a
three-layer network (i.e., with a single hidden layer) is selected in this study to investigate the relative importance of different subsets of echo features on the performance of a given network for the classification of seafloor sediments. The output layer consists of two neurons. Two neurons are sufficient to classify four types of sediments (such as [00], [01], [10], and [11]). The number of neurons in the hidden layer is safely chosen as 20. The number of neurons in the input layer varies from 2 to 7. Thus, 6 network configurations such as [2-20-2] (i.e., 2 input neurons, 20 hidden neurons, and 2 output neurons), [3-20-2], [4-20-2], [5-20-2], [6-20-2], and [7-20-2] are considered in this study.

Initially, few trials are made to assess the consistencies of the success rates of these 6 network configurations with varying number of hidden neurons ranging from 5 to 30. It is observed that 12 to 14 neurons in the hidden layer are necessary to achieve the optimized result when the number of neurons in the input layer changes from 2 to 4. Similarly, it is also observed that 16 to 20 neurons in the hidden layer are required to achieve the consistency in the success rate using different subsets of echo features when the input neuron number changes from 5 to 7. The increase in the number of neurons in the hidden layer (beyond an optimized value for a given network configuration) does not increase the success rate in the classification, however, the computational time marginally increases. Since the aim of this study is to select an optimum feature subset using MLP networks, the number of neurons in the hidden layer is safely chosen as 20 (which is sufficient to produce consistent results using all the 6 network configurations considered in this study).

Sigmoidal transfer function such as tan sigmoid is used as an activation function in each layer of a three-layer network. Resilient backpropagation algorithm is used as training algorithm of the MLP network. The Neural Network Toolbox available in
MATLAB 7.0 (2004) is used in this study. Various training parameters are: learning rate - 0.05; increment in weight change - 1.2; decrement in weight change - 0.5; initial weight change - 0.07; maximum weight change - 50; and epochs - 6000. A fixed 15% of the total input dataset (consisting of 7 echo features) is optimally used as training dataset and the remaining 85% of the total dataset is used as a validation (or testing) dataset.

It is mentioned that the performance of a neural network for classification of seafloor sediments greatly depends on the training dataset. Significant performance can be achieved if suitable echo features with sufficient discriminating characteristics are chosen optimally for training a neural network. A series of experiments are carried out with different subsets of echo features as inputs to the three-layer MLP network. Different subsets of echo features consist of 2, 3, 4, 5, and 6 features (taken at a time out of the 7 echo features). The overall percentages of success to classify seafloor sediments are evaluated with all these subsets. Accordingly, when a subset with 2 echo features is used as an input, the network configuration [2-20-2] is used. Similarly, [3-20-2], [4-20-2], [5-20-2], and [6-20-2] network configurations are used when the input feature subsets consist of 3, 4, 5, and 6 features. As mentioned earlier, the number of neurons in the hidden layer is kept at a fixed value to assess the performance of a network for different subsets of input features. If 2 or 5 different echo features are selected at a time (without regard to order) out of 7 features, then there exist 21 subsets of inputs (from binomial coefficient). Similarly, there exist 35 subsets of inputs if 3 or 4 different echo features are selected at a time out of 7; and 7 subsets if 6 different echo features are chosen together.

The performance (percentage success of correct classification during testing) of the network [2-20-2] is evaluated with a subset of 2 echo features (for example BS and
TS) for all sand, silty sand, silt, and clayey silt sediments at twenty locations. Subsequently, an average percentage of success of the network is computed by taking a mean value of the successes at four sedimentary environments. However, this average percentage of success of the network depends on the initial values of the interconnecting weights (i.e., initialization of the weights during the training of a network). Therefore, the network [2-20-2] is trained and tested 10 times with 10 different sets of initial weights. The number 10 is chosen arbitrarily and it is used to take into account the effect of the variations of initial weights (which are chosen randomly during the training process) on the performances of a network. Subsequently, an overall average percentage of success is obtained by taking the average of these 10 sets of results (for the above-said two features, BS and TS). Similarly, the same procedure is followed for the rest of the 20 subsets of input features (such as [BS, HD], [TS, HD], [StatSkew, HD] etc.) to compute the overall average percentages of success for the network [2-20-2]. Likewise, the overall average percentages of success are computed for all the 21 subsets of input features using the network [5-20-2]. Similarly, overall average percentages of success are computed for the 35 subsets using the networks [3-20-2] as well as [4-20-2], and 7 subsets for the network [6-20-2].

6.3.3 Results and Discussion

The overall average percentages of success thus obtained from all the 5 networks (i.e., [2-20-2], [3-20-2], [4-20-2], [5-20-2], and [6-20-2]) are first sorted in ascending order (separately for each network) and plotted in Fig. 6.9 for 33 kHz and Fig. 6.10 for 210 kHz. The sequence numbers along the x-axes of Fig. 6.9 and Fig. 6.10 essentially indicate different subsets of echo features. However, a particular sequence number for
[2-20-2] network and for [5-20-2] network does not represent the same feature subset (as [2-20-2] network has 2 input features and [5-20-2] network has 5 input features). These plots clearly indicate how the performances of a network vary with different subsets of input features, if all other parameters remain unchanged. The following observations are obtained from these experiments.

1. The overall average percentage of success in classifying seafloor sediments with a MLP network is reasonably higher for 210 kHz than that for 33 kHz.

2. More numbers of echo features are required for achieving higher success rate at 33 kHz. A subset of two or three echo features as an input to the network could not produce a maximum success rate beyond 89% at 33 kHz. However, subsets of 4, 5, and 6 echo features as inputs to the network produce maximum success rate nearly 92-93% at 33 kHz. On the other hand, the maximum overall average percentages of success lie within 97-98.5% for all subsets of input features at 210 kHz.

This observation can be explained in terms of penetration of the acoustic energies into the seafloor sediments. The seafloor interface scattering component dominates (compared with the lesser sediment volume scattering component) in the total backscatter strength at 210 kHz because of the weaker penetration of acoustic energies into the sediment volume. On the contrary, 33 kHz penetrates the sediment volume at a relatively greater depth compared to 210 kHz and thus the sediment volume scattering component (in addition to the seafloor interface scattering component) has a significant effect on the total backscatter strength. It indicates that more numbers of echo features are necessary to discriminate different types of surficial seafloor sediments because of the significant contribution of volume scattering component at 33 kHz.
3. It is observed that if all echo features are used as input to a MLP-based classifier (i.e., using [7-20-2] network configuration), the overall percentages of success for seafloor classification are 91.19±1.73 and 95.43±2.39 respectively for 33 and 210 kHz. Therefore, maximum performance of the MLP network degrades marginally with the use of more than optimum number of echo features at both the acoustic frequencies. The neural network configuration [4-20-2] gives highest performance at 33 and 210 kHz. However, the lowest limit of performance for a given network increases with the increase in the number of features as inputs (as shown in Table 6.1).

![Fig. 6.9](image-url)  
**Fig. 6.9**  
Showing the results of overall average percentage of success obtained with different subsets of input features at 33 kHz. Each sequence number along the x-axis represents different feature subsets.
4. The results show that the highest performance (based on the overall average percentage of success) of the neural network based sediment classifier is achieved with a subset of four features consisting of BS, TS, StatSkew, and HD. In addition, the results show that the percentage increase in the success from three-features input subset to four-features input subset is marginal for 210 kHz as compared to 33 kHz. The feature subsets namely [SpKurt, StatSkew] and [SpSkew, SpKurt, StatSkew] give very low (≤50%) success at 33 kHz. In contrast, the feature subsets namely [SpSkew, SpWidth], [SpWidth, HD], and [SpSkew, SpKurt, SpWidth] give very low (≤50%) success at 210 kHz for seafloor classifications. In addition, it is noticed that training of a neural network does not converge successfully if these feature subsets are used.

Fig. 6.10  Showing the results of overall average percentage of success obtained with different subsets of input features at 210 kHz. Each sequence number along the x-axis represents different feature subsets.
Table 6.1 Showing the results with highest and lowest overall average percentages of success for seafloor classification with MLP networks. Here F1, F2, ..., F7 represent backscatter strength, spectral skewness, spectral kurtosis, spectral width, statistical time-spread, statistical skewness, and Hausdroff dimension respectively.

<table>
<thead>
<tr>
<th>Acoustic Freq</th>
<th>Network Configuration</th>
<th>% Success ± STD*</th>
<th>Feature Set</th>
<th>Time (s) ± STD*</th>
<th>% Success ± STD*</th>
<th>Feature Set</th>
<th>Time (s) ± STD*</th>
</tr>
</thead>
<tbody>
<tr>
<td>33 kHz</td>
<td>[2-20-2]</td>
<td>83.9 ± 1.9</td>
<td>(F1-F5)</td>
<td>101.2 ± 2.1</td>
<td>46.9 ± 3.8</td>
<td>(F3-F6)</td>
<td>102.8 ± 3.1</td>
</tr>
<tr>
<td></td>
<td>[3-20-2]</td>
<td>88.7 ± 1.1</td>
<td>(F1-F5-F6)</td>
<td>103.3 ± 1.8</td>
<td>46.9 ± 1.7</td>
<td>(F2-F3-F6)</td>
<td>104.3 ± 2.2</td>
</tr>
<tr>
<td></td>
<td>[4-20-2]</td>
<td>92.8 ± 1.3</td>
<td>(F1-F5-F6-F7)</td>
<td>75.0 ± 2.2</td>
<td>52.2 ± 2.9</td>
<td>(F2-F3-F6-F7)</td>
<td>114.4 ± 2.0</td>
</tr>
<tr>
<td></td>
<td>[5-20-2]</td>
<td>92.5 ± 1.5</td>
<td>(F1-F3-F5-F6-F7)</td>
<td>35.7 ± 3.1</td>
<td>66.3 ± 2.6</td>
<td>(F2-F3-F4-F6-F7)</td>
<td>115.3 ± 1.1</td>
</tr>
<tr>
<td></td>
<td>[6-20-2]</td>
<td>92.2 ± 1.1</td>
<td>(F1-F2-F4-F5-F6-F7)</td>
<td>20.4 ± 5.4</td>
<td>81.9 ± 3.6</td>
<td>(F2-F3-F4-F5-F6-F7)</td>
<td>124.8 ± 4.2</td>
</tr>
<tr>
<td>210 kHz</td>
<td>[2-20-2]</td>
<td>97.3 ± 1.0</td>
<td>(F1-F7)</td>
<td>9.4 ± 1.6</td>
<td>48.9 ± 2.6</td>
<td>(F2-F4) &amp; (F4-F7)</td>
<td>79.5 ± 2.1</td>
</tr>
<tr>
<td></td>
<td>[3-20-2]</td>
<td>97.8 ± 1.1</td>
<td>(F1-F5-F7)</td>
<td>6.3 ± 0.8</td>
<td>49.7 ± 2.1</td>
<td>(F2-F3-F4)</td>
<td>78.6 ± 1.7</td>
</tr>
<tr>
<td></td>
<td>[4-20-2]</td>
<td>98.4 ± 0.6</td>
<td>(F1-F5-F6-F7)</td>
<td>6.0 ± 0.9</td>
<td>54.7 ± 2.0</td>
<td>(F2-F3-F4-F7)</td>
<td>83.6 ± 1.0</td>
</tr>
<tr>
<td></td>
<td>[5-20-2]</td>
<td>97.6 ± 1.3</td>
<td>(F1-F3-F5-F6-F7)</td>
<td>5.9 ± 1.2</td>
<td>58.8 ± 1.1</td>
<td>(F2-F3-F4-F6-F7)</td>
<td>83.5 ± 1.4</td>
</tr>
<tr>
<td></td>
<td>[6-20-2]</td>
<td>97.1 ± 1.0</td>
<td>(F1-F2-F3-F4-F5-F6-F7)</td>
<td>5.8 ± 1.2</td>
<td>80.4 ± 3.9</td>
<td>(F2-F3-F4-F5-F6-F7)</td>
<td>40.4 ± 5.1</td>
</tr>
</tbody>
</table>

* STD - Standard Deviation values

5. Moreover, the results reveal that the overall average percentage of success decreases drastically (as shown in Fig. 6.9 and Fig. 6.10) after a certain point for all the networks at both the acoustic frequencies 33 and 210 kHz. The results for 33 kHz reveal that if a feature subset contains BS or TS as one of the inputs, seafloor classification successes are higher than 73%, 73%, 66%, and 76% with [2-20-2], [3-20-2], [4-20-2], and [5-20-2] networks respectively. In addition, if BS is one of the input features in a subset, seafloor classification success is higher than 88% for [6-20-2] network at 33 kHz. It is also observed that if the subset of features contains backscatter strength (BS), the overall average percentages of success of a network are above 89% for 210 kHz. This interesting aspect is noticed uniformly in all the cases as shown in Fig. 6.9 and Fig. 6.10. The highest and the lowest success rates (with standard deviation i.e., STD values) are listed in Table 6.1 along with the respective feature subsets. This helps in understanding the relative importance of a feature subset for achieving the higher success in seafloor classification. This is to
mention that all the computations are carried out using MATLAB 7.0 (2004), which is installed in a computer having AMD Athlon 64 bit 3000+ processor with 512 MB RAM. The average computational times with standard deviation values are also shown in Table 6.1 for the different network configurations.

6.3.4 Conclusions

This study demonstrates that the improved performance of MLP networks based sediment classifier can be achieved if the input echo features are selected preferentially. The use of more than the optimum number of echo features in a given neural network does not necessarily increase the success rate of a neural classifier. The analysis reveals that backscatter strength, time spread, statistical skewness, and Hausdroff dimension are the most discriminatory echo features (when used as a subset) to MLP neural networks for the classification of seafloor sediments at 33 and 210 kHz. The analyses also reveal that backscatter strength is the most dominant echo feature at both the acoustic frequencies, while time spread is important (in addition to backscatter strength) at 33 kHz only. In addition, the study reveals that 210 kHz is advantageous (in comparison with 33 kHz) for the classification of seafloor sediments. However, the limitation of supervised neural networks based classifier is that these methods require a-priori knowledge (i.e., the ground-truth) on sediment types available in the study area to decide the output target vectors for training such networks.

This chapter has proposed a supervised method to select the prominent echo features using multilayer backpropagation networks. A hybrid approach using Kohonen’s self-organizing feature map and Fuzzy C-Means cluster algorithm is proposed in the next chapter.