Better Learning of Supervised Neural Networks Based on Functional Graph – An Experimental Approach

JOSEPH RAJ V.
Department of Computer Engineering
Faculty of Architecture and Engineering
European University of Lefke
TRNC, Mersin 10
TURKEY
v.jose08@gmail.com

Abstract: Multilayered feed forward neural networks possess a number of properties which make them particularly suited to complex problems. Neural networks have been in use in numerous meteorological applications including weather forecasting. As Neural Networks are being more and more widely used in recent years, the need for their more formal definition becomes increasingly apparent. This paper presents a novel architecture of neural network models using the functional graph. The neural network creates a graph representation by dynamically allocating nodes to code local form attributes and establishing arcs to link them. The application of functional graph in the architecture of Electronic neural network and Opto-electronic neural network is detailed with experimental results. Learning is defined in terms of functional graph. The proposed architectures are applied in weather forecasting and X-OR problem. The weather forecasting has been carried out based on various factors consolidated from meteorological experts and documents. The inputs are temperature, air pressure, humidity, cloudiness, precipitation, wind direction, wind speed, etc., and outputs are heavy rain, moderate rain and no rain. The percentage of correctness of the weather forecasting of the conventional neural network models, functional graph based neural network models and the meteorological experts are compared.

Keywords:- Back propagation, Convergence, Functional Graph, Neural network, Optical neuron, Weather forecasting

1 Introduction

Rain is one of nature's greatest gifts to mankind. It is a major concern to identify any trends in rainfall deviation from its periodicity, which would disrupt the economy of a country. In the present study weather forecasting (rainfall patterns) is carried out based on temperature, air pressure, humidity, cloudiness, precipitation, wind direction, wind speed, etc., consolidated from meteorological experts and documents [1].

Artificial Neural Network (ANN) can be defined as a model of reasoning based on the human brain. Recent developments on ANN have been used widely in weather forecasting because of its ability to generalize well on the unseen patterns. The neural network creates a graph representation by dynamically allocating nodes to code local form attributes and establishing arcs to link them. All the models define ANN's only as static objects where no change in parameters is done and the learning phase is only said to have taken place previously, or it defined using a different formalism than the one describing the ANN's architecture [2]. There seems to be a need for an ANN model, where all its phases as well as ANN's architecture, are defined using the same formalism. Functional graphs seem to be perfect for this purpose. A functional graph is a digraph describing the action of a function on a set. Functional graph is a system of nodes connected with functional edges between which binary relations can be defined. Multilayer artificial neural networks can be easily defined using functional edges to model neurons, and parameterized binary relation to model connections.

In this paper an attempt has been made to analyze the performance of functional graph implementation in conventional electronic neural network model and opto-electronic neural network model. It is experimented with weather forecasting and X-OR problem. The paper concludes with a section on performance comparisons and considers the future possibilities. The Functional Graph based neural network models converge quickly compared to conventional ANN models in weather forecasting.

2 Statement of the Problem

The motivations of this paper are

(i) to study the effect of the electronic neural network model based on the functional graph implementation
(ii) to study the effect of the opto-electronic neural network model based on the functional graph implementation
(iii) to study the effect of functional graph based neural network models on X-OR problem and
(iv) to study the effective factors of weather forecasting.

3 Artificial Neural Networks

3.1 Electronic Neural Network

The two important issues in supervised learning are performance and efficiency [3]. In a supervised learning environment, let $D = \{x_n, t_n\}_{n=1}^{N}$ denote N training samples pairs, where $x_n \in R^d$ is a d-dimensional feature vector of the n th sample and t_n is the corresponding target. Furthermore, we assume that there is a function $f(\cdot)$ so that $t_n = f(x)$. For the function approximation problem, t_n is a real number. For classification, t_n indicates the class to which the n th sample belongs. Neural networks are a class of nonlinear models that consist of interconnected nonlinear processing nodes. Let $W^{(1)}$ be a weight matrix at the first layer and its j th column $w_j^{(1)}(1 \le j \le L)$ be the weight vector connecting the j th hidden unit to the inputs. Let $w^{(2)}$ be a weight vector at the second layer and its j th element $w_j^{(2)}$ denote the weight connecting the j th hidden unit to the output. h_n is a vector that represents the outputs of hidden units corresponding to input x_n. The j th element $h_{j,n}(1 \le j \le L)$ of h_n is equal to

$$h_{j,n} = g\left(w_j^{(1)T} x_n\right) \tag{1}$$

where $g(\cdot)$ is the sigmoidal transfer function. The output of the network, y_n, corresponding to an input x_n can be expressed as

$$y_n = w^{(2)T} h_n \tag{2}$$

Let $f_D(x; w)$ be a model with a set of parameters w and trained on training set D. The performance of $f_D(x; w)$ can be measured in terms of the difference between a function $f(x)$ to be approximated and its approximation $f_D(x; w)$ through the squared norm

$$\int \left| f(x) - f_D(x; w) \right|^2 p(x) dx \tag{3}$$

where $p(x)$ is the probability density function of x. If training samples are drawn randomly, the expected squared norm is

$$E_D\left(\left\| f(x) - f_D(x; w) \right\|^2 \right) = E_D\left(\int \left| f(x) - f_D(x; w) \right|^2 p(x) dx \right) \tag{4}$$

where the expectation (E) is over all possible training sets D of the same size. The quantity $E_D\left(\left\| f(x) - f_D(x; w) \right\|^2 \right)$ can be considered as a measure of the performance. It is also called the generalization error of $f_D(x; w)$, since it measures the average performance of $f_D(x; w)$ in approximating the unknown function $f(x)$ rather than fitting N training samples alone. When pattern classification is considered as a special case of nonlinear regression, generalization error can be defined as the probability of incorrect classification, which is $E_D\left(\Pr\left(f_D(x; w) \ne t\right)\right)$.

The space complexity S of $f_D(x; w)$ is the number of free parameters l of $f_D(x; w)$ when the generalization error is no bigger than $\varepsilon_g > 0$. For example, if a three-layer feed forward neural network is considered, l can be either the number of hidden units or the total number of independent weights of the network. The time complexity of an adaptive learning system is defined as the expected training time needed to obtain a learning system when the generalization error is bounded by ε_g. The scaling property of the time complexity and space complexity defines the efficiency of an adaptive learning machine (or algorithm) whose generalization error is no bigger than a given quantity, $\varepsilon_g > 0$. $f_D(x; w)$ is said to be efficient in space if S scales polynomially in terms of the dimension d of feature vectors, ε_g, and other

related parameters. If the time complexity scales polynomially in term of these quantities, the learning machine is said to be efficient in time. If the efficiency can be achieved both in space and time, the learning machine is said to be efficient.

The generalization error is affected by two factors bias and variance. The error due to the bias that can be caused by an inappropriate choice of the size of a class of models when the number of training samples is assumed to be infinite. The error due to variance is caused by the finite number of training samples. In a special case when a class of models was chosen to be three-layer feedforward neural networks with L sigmoid hidden units, the bias and variance were bounded explicitly by Barron [4].

One popular general heuristic technique is that of neural networks. If simulated annealing appeals to physicists and genetic algorithms appeal to biologists, neural networks appeal to brain surgeons. By drawing an analogy with how the brain stores and processes information, neural networks attempt to mimic the brain's flexibility in solving problems. Electronic Neural network (ENN) is a set of very simple processing elements (neurons) with a high degree of interconnections (weights) between them. Such processing architecture is characterized by parallel processing and distributed computing, and is able to learn and adapt to better perform a specific task by modifying the strength of connections between the neurons [5,6] and shown in Fig.1.

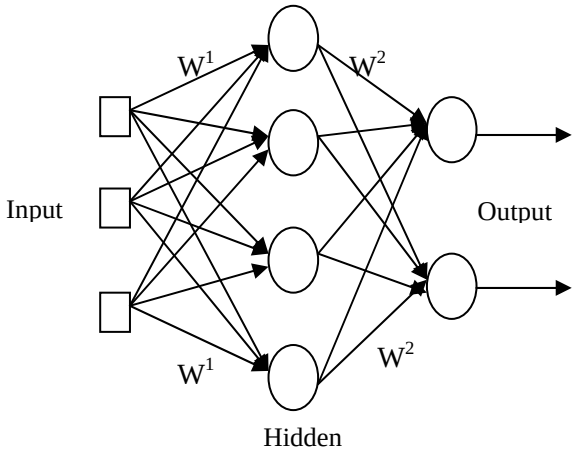

Fig.1 Multilayer electronic neural network

3.2 Opto–electronic Neural Network

Optics can play an imponant role in overcoming the bottlenecks imposed on purely electronic systems by the power and space demands of connections between chips. Optical architectures can achieve a high density of interconnection more simply by making use of the third Dimension; electronic structures are essentially two dimensional, albeit with multiple levels. Neural networks, because of their dependence on dense interconnection, have the most to gain from the inclusion of optics and many workers have developed opto-electronic implementations.

The optical neuron, modeled using the following relations, is used in the neurons of Opto-electronic Neural Network (ONN) [7,8]. The ONN network, multilayer perceptron, trained using modified back propagation is constructed as three layer network with one hidden layer.

Light input to photo detector:

$$Linput_{ij} = Loutput_i * optical\_attenuation \quad (5)$$

Photo detector photo current:

$$Iph_{ij} = \gamma_p * V_{bias\,ij} * Linput_{ij} \quad (6)$$

where $Linput_{ij}$ is the light input of neuron j in the hidden layer from the neuron i, in the input layer

Amplifier output :

$$I_j = \sum Iph_{ij} * \left[1/\left(\left|\gamma_p\right| * optical\_attenuation\right)\right] \quad (7)$$

Sigmoid output :

$$Is_i g_j = I_{\max} / \left(1 + e^{-I_J / I_{\max}}\right) \quad (8)$$

Current output :

$$Ioutput_j = 1/\left|\gamma_l\right| * Is_i g_j \quad (9)$$

Light output:

$$Loutput_j = \gamma_l * Ioutput_j \quad (10)$$

Input layer output:

$$Loutput_i = Input_i * \left(1 + noisefactor * randomnumber\right) \quad (11)$$

where $Input_i$ is the input to the input layer

Most units in neural network transform their net input by using a scalar-to-scalar function called an "activation function", yielding a value called the unit's "activation". Sigmoid function's output is, in the range between [0,1]. Since sigmoid function is output limiting and quasi bi-stable, it is also differentiable.

$$Sigmoid\_function = 1/\left(1+e^{-x}\right) \quad (12)$$

where x is the input to sigmoid function

4 Graphs - A Survey

The application of Graphs is becoming popular in modeling neural networks and solving pattern classification problems. Yanjing Sun, et.al. [9] propose a distributed algorithm for finding an approximation of a minimum connected dominating sets to construct a virtual backbone in the growth-bounded graph. The growth-bounded graph captures the intuitive notion that if many nodes are located close from each other, many of them must be within mutual transmission range. In graph theoretical terms, a graph is growth-bounded if the number of independent nodes in a node's r-neighborhood is bounded.

Different parallel architectures may require different algorithms to make the existent algorithms on one architecture be easily transformed to or implemented on another architecture. Shih-Jung Wu an, et.al. [10] propose an algorithm for embedding complete binary trees in a faulty Incrementally Extensible Hyercube (IEH). A hypercube Q_n of order n, is defined to be a symmetric graph $G=(V,E)$ where V is the set of 2^n vertices, each representing a distinct n-bit binary number and E is the set of symmetric edges such that two nodes are connected by an edge if and only if (iff) the number of positions where the bits differ in the binary labels of the two nodes is 1.

The IEH graph is the composition of some m different hypercubes. Let $G_n(N)$ be an IEH graph with N nodes, and N can be expressed by the binary string $N=b_nb_{n-1}b_{n-2}\cdots b_1b_0$, and $b_i\in\{0,1\}$. An IEH graph $G_n(N)$ is composed of some different hypercubes which have lower dimension than $G_n(N)$ has. That is, $G_n(N)$ contains a hypercube, denoted by H_i, if and only if the i th bit in the binary representation of N is 1. Accordingly, the IEH graph is composed of some hypercubes, so there is a new type of connections beside the usual connections in a hypercube. These edges (or links) which are used for connecting two hypercubes are called Inter-Cube or IC edges. The basic philosophy in the design of the IEH graphs is to express N as a sum of several powers of 2, i.e., to write N as a binary number, build the smaller hypercubes, and then to add appropriate inter-cube edges to connect those smaller hypercubes.

Wesam Barbakh, et.al. [11] introduce a new algorithm to build the connectivity graphs. To construct a connectivity graph we transform a given set $x_1,...,x_n$ of data points with pair wise similarities s_{ij} or distances d_{ij} into a graph. There are several popular methods to construct similarity graphs. The goal of constructing connectivity graphs is to model the local neighborhood relationships between data points.

Larbi Sekhri, et.al. model [12] follow the terminology and the notations of Petri nets. It is used for pattern recognition.

Definition 1. A Petri net N is a 4-tuple $G=(P,T,F,V)$ where $G=(P,T,F,V)$ is a weighted bipartite digraph. P is a finite set of state places and T is a finite set of transitions with $P\cup T\neq\phi$ and $P\cap T=\phi$. F is a set of directed arcs connecting state places to transitions and vice-versa. V is the weight application (valuation): $V\in\left[F\rightarrow N^+\right]$.

Definition 2. A marking M of a Petri net $G=(P,T,F,V)$ is a mapping from P to N (set of integer) where $M(p)$ denotes the number of tokens contained in place p. A marked Petri net is a couple (G,M_0) where G is a Petri net and M_0 a marking of G called the initial marking. We denote for a node $s\in P\cup T, \bullet s (resp. s\bullet)$ the set of nodes such that $(s',s)\in F\left(resp.(s,s')\in F\right)$. The valuation V of a Petri net can be extended to the application W from $(P\times T)\cup(T\times P)\rightarrow N$ defined by: $\forall u\in(P\times T)\cup(T\times P), W(u)=V(u)$ if $u\in F$ and $W(u)=0$ otherwise. The matrix C indexed by $P\times T$ and defined by $C(p,t)=W(t,p)-W(p,t)$ is called the incidence matrix of the net.

Definifion 3. Let (G,M_0) be a marked Petri net. A transition $t\in T$ is enabled by a marking M if and only if $\forall p\in\bullet t, M(p)\geq W(p,t)$. The marking M' reached by firing t at M is defined by

$$\forall p\in P, M'(p)=M(p)-W(p,t)+W(t,p)$$

denoted by $M\ [t>M'$.

By extension, a marking M' is said to be reachable from a marking M if there exists a sequence of transitions $s=t_0t_1...t_n$ and a series of marking $M_1,...,M_n$ such that

$$M\left[t_0>M_1, M_1\left[t_1>M_2,...,M_n\left[t_n>M'\right.\right.\right. \quad (denoted$$

$M[s > M'\rangle$. The set of the markings of N reachable from a marking M is denoted by $R(G, M)$.

Definition 4. Let (G, M_0) be a marked Petri net.

A transition t of G is live if and only if $\forall M \in R(G, M_0), \exists M' \in R(G, M)$ such that $M'[t>$. (G, M_0) is quasi-live if and only if $\forall t \in T, \exists M \in R(G, M_0)$ such that $M[t>$. (G, M_0) is weakly-live (deadlock-free) if and only if $\forall M \in R(G, M_0), \exists t \in T$ such that $M[t>$. (G, M_0) is called bounded if and only if $\exists k \in N / \forall M \in R(G, M_0), \forall p \in P, M(p) \leq k$.

(G, M_0) is live if and only if all transitions of G are live.

G is structurally live if and only if $\exists M_0 \in N^p$ such that (G, M_0) is live.

Functional Nets are a kind of Petri net models obtained by the translation of Functional Graphs. The aim is to define a systematic transcription that enables us to capture in the Functional nets the causal properly of a Functional Graph.

Definition 5. A Functional Net is an ordinary Petri net which models a Functional Graph.

The diagnosability analysis technique starts with the translation of Functional graphs into Petri net models and uses reachability graph of these specific Petri nets. The study distinguishes three kinds of basic motifs in a Functional graph.

1. Motif I (a function with one mother function)
2. Motif A (convergent function with two daughter functions)
3. Motif V (divergent function with two mother functions)

Each Functional graph can be represented as a combination of these three motifs. The process of Functional net generation uses three mapping rules based upon the propositions

1. Motif rule I
2. Motif rule A
3. Motif rule V

These motif rules are associated respectively to the preceding basic motifs. Indeed, the process of Functional net generation uses a marking rule. Decreasing priority order between motifs is given as follows: Motif I, Motif A and Motif V. This priority between motifs enables to recognize fragments (motifs) in Functional graphs in an efficient and deterministic manner.

Rule I. Motif I (Exclusive connection between two adjacent functions)

The Functional net translation maps the Motif I into the Functional subnet (Petri net). Places represent the functions of the initial functional graph transitions model causality relations between its functions.

Rule A. MotifA (Convergent function)

Function fl is diagnosable if the daughter functions f2 and t3 are diagnosable. This kind of Functional graph is translated into a Petri net composed by three places and a synchronization transition tl. The place pl becomes marked if places p2 and p3 are marked.

Rule V. Motif V (Divergent function)

The diagnosability of the divergent function fl in the Functional graph is propagated by causality to mother functions f2 and f3. This Functional graph is translated into the Functional net. If pl is marked (fl is diagnosable) then the places p2 and p3 become marked. This means that functions f2 and f3 are diagnosable. If f2 or f3 is diagnosable f1 becomes diagnosable by causality. This causality relation is provided by the transition t2 or t3.

Remark:

The causality of the diagnosability is propagated from f1 to functions f2 and f3 only if these functions are not convergent functions.

Marking rule:

A marked place means that the corresponding function is diagnosable. The marking rule is related to the initial marking of places in a Functional net. The initial marking corresponds to functions that are directly diagnosable by sensors. So a Functional net is diagnosable if all its places can be marked at least once.

A Functional Graph is a graphical model that explains how the functions of the system components are combined to implement the services it delivers [13]. It is composed of nodes that model each of these functions. Directed arrows that model functional dependency relationships interconnect these functions. The dependency relationships are of different types: composition relationships, flow relationships or adaptation relationships. In the functional hierarchy given by the Functional Graph, some functions are distinguished: Principal Functions that corresponds to the system services and Initial Functions. Initial functions are implemented by the basic components of the process. Consequently they are the initial causes of every faults of the system.

Graphically, the model is also completed by the addition of "or" operators (noted by a "+") that enable the designer to model the cases of functional redundancy. A functional redundancy is the fact that higher-level

function can be implemented differently by several lower level functions. In the other cases, the functional dependence relationships are constrained by an "and" operator. It is sufficient that one function among f2, f3 and f4 becomes unavailable to imply that f1 becomes unavailable too. At the opposite the three functions are connected by an "OR" operator. In this case, it is possible to ensure the availability of f1 with only one of its sub-functions. There is also a more complicated representation of functional relationships constrained by "AND-OR" connections.

5 Functional Graph

5.1 Functional Graph - Introduction

Functional graph [14] is a system S = (X, Y, F, R) where X and Y are set of inputs and outputs of the system elements

$$X = \bigcup_{i=1}^{m} X_i \tag{13}$$

$$Y = \bigcup_{i=1}^{m} Y_i \tag{14}$$

$$X_i = \{x_{i1},...,x_{i\alpha i}\} \tag{15}$$

$$Y_i = \{y_{i1},...,y_{i\alpha i}\} \tag{16}$$

where F is the set of elements f_i called functional edges which are pairs

$$f_i = (X_i, Y_i), F = \{f_i : i = 1,2,...,m\} \tag{17}$$

R is a set of binary relations $R \subseteq Y \times X$. The relation

$$R_i = \{(y,x) : (y,x) \in R, x \in X_i\} \tag{18}$$

is called a connecting edge. A functional interpretation, which gives rise to systems with functional interpretation, is also given. With every functional edge f_j an operator is associated with

$$W_j : \{P\} \times (x_1 \times ... \times x_n) \to (y_1 \times ... \times y_m) \tag{19}$$

where P is a set of parameters. The relation R is attributed with the meaning that if (y, x) $\in$ R, then the values of x and y must be equal.

5.2 Functional Graph Model

Functional graphs can be used to establish a model of a neural network. Let it be a model of a single hidden layer feed forward neural network with m inputs, n hidden nodes, and p outputs. A neuron node will be represented with a functional edge

$$f_i^{lay} \in F = \{F^{lay}\}_{lay} \in LAYERS \tag{20}$$

$$f_i^{lay} = (X_i^{lay}, ACT_i^{lay}) \tag{21}$$

where lay is the layer identifier

$$lay \in LAYERS = \{in, hid, out\} \tag{22}$$

$ACT_i^{lay} \in IR$ is a set of possible activations of neuron i in layer lay. There is a mapping

$$actfun : F^{lay} \to FA = \{fact_j\} \tag{23}$$

where

$$fact_j : W_j^{lay} \times lay \times (x_1,..., x_{num(lay)}) \to ACT_j^{lay} \in IR \tag{24}$$

where num(lay) is an operator which gives the number of nodes in layer lay. W_j^{lay} is a num(lay) dimensional vector of parameters representing the synaptic connection weights incoming into node j from previous layer nodes. actfun simply associates with a node a method of computation and an activation function.

For example, in a feedforward network given in Fig.2, actfun will associate with neuron node f_1^{hid} the function

$$actfun(f_1^{hid}) = f\left(\sum_{i=1}^{3} Wi^{hid} X_i\right) \tag{25}$$

If the network is to use the backpropagation algorithm, then f shall be the sigmoidal function $f : 1/(1+e^{-x})$ where as, if as the network teaching algorithm the steepest descent method were used and the neuron is linear, then

$$f : IR \ni x \to x \in IR \tag{26}$$

or, if nodes inputs and activations are discrete, then the activation function can take the following form

$$f(x) = \begin{cases} +1; x > 0 \\ 0; x = 0 \\ -1; x < 0 \end{cases} \tag{27}$$

Thus the definition of a neuron node as a functional edge does not define all codes as identical; every node can have its own, different, computation method. The method can even change through the learning process.

W_{ji}^{lay} will represent the synaptic connection weight between neuron i in layer directly preceding layer lay with neuron j in layer lay.

- W_j^{lay} a vector of weights incoming into node j in layer lay.
- W^{lay} a vector of weights incoming into all neuron nodes in layer lay.
- W a vector of all weights in the system

$$W_{ji}^{lay} : x \to W_{ji}^{lay} x \tag{28}$$

$$W_j^{lay} = \left\{ W_{ji}^{lay} : i = 1,..., num\left(pred\left(lay\right)\right) \right\} \tag{29}$$

$$W^{lay} = \left\{ W_j^{lay} : j = 1,..., num\left(lay\right) \right\} \tag{30}$$

$$W = \left\{ W^{lay} \right\}_{lay} \in LAYERS \tag{31}$$

where pred(lay) gives the identifier of layer directly preceding lay.

The connecting edges P_{ji}^{lay} will represent the existence of a synaptic connection between a pair of nodes j in layer lay and i in the directly preceding layer. The associated meaning would be as follows: if relation yP_{ji}^{lay} x holds (i.e., there is a synaptic connection), then y = x (the activation value of neuron i is passed on); if it does not (i.e., there is no connection), then y = 0.

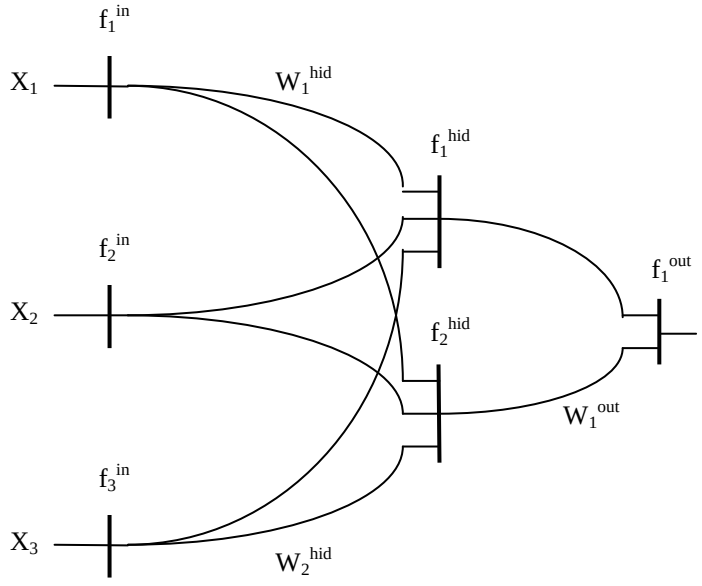

Fig.2 Model of ANN as a functional graph

5.3 Learning with Functional Graph

Learning is an essential attribute of neural networks. There is a number of learning algorithms which are of either supervised type, when desired answers are given with the learning examples, or of unsupervised type when answers are not given; algorithms can be of one-shot type, when each example is examined only once during the whole training, or of iterative type when examples are cycled through many times until a given condition, usually a minimal value of an error function, is met. Algorithms also depend on the type of architecture of the artificial neural network used.

For the feedforward networks one of the most widely used is the gradient descent method. Having a differentiable error function E that takes a vector of weights

$$W = \left(w_0, w_1,..., w_p\right) \tag{32}$$

E(W) which gives the error value for these weights, and a given example, one can compute its multidimensional gradient vector

$$\nabla E = \left(\partial e / \partial w_0 ... \partial e / \partial w_p\right) \tag{33}$$

The negative gradient vector represents in weight space the direction which would result in error decrease if a minimal weight change took place in that direction

$$W^* = W - \eta \nabla E\left(W\right) \tag{34}$$

where W* is the new weight vector, and η is the learning rate parameter.

Using gradient descent method and functional graph, the networks constructed for weather forecasting during the several stages of network architecture development using input and output necessitated by the problem domain with three layer networks and one hidden layer.

6 Methodology

The basic input data for classification needs preprocessing [15,16]. This procedure helps in the incorporation of temperature, air pressure, humidity, cloudiness, precipitation, wind direction, wind speed, etc., in weather forecasting for ANN models and Functional graph ANN models based classification. The factors: temperature, air pressure, humidity, cloudiness, precipitation, wind direction, wind speed, etc., of weather forecasting are consolidated from meteorological experts and documents. The inputs to the ANN models and Functional Graph based ANN models are, in the form of 1 or 0 based on the presence or absence of the factors. In this study, the different types of weather forecasting (rainfall patterns) are heavy rain, moderate rain and no rain. The output of the ANN models and Functional Graph based ANN models are in the form of 1 or 0, based on presence or absence of the types of weather forecasting.

The ENN, Functional Graph based ENN, Opto-electronic neural network and Functional graph based opto-electronic neural network are trained with twenty seven inputs (factors) and three outputs. The neural network models are trained with samples of patterns collected from Meteorological Department of Kanyakumari District of Tamilnadu, India. The number of inputs is limited and outputs are selected based on the advice of meteorological experts and documents.

7 Results and Discussion

Improvement of classification accuracy in weather forecasting is an important issue. The trained neural network models are tested with samples and validated for the accuracy of the models. The test data are analyzed with two meteorological experts for its accuracy. The percentage of correctness of weather forecasting is tabulated in Table 1 and shown in Fig.3.

If the generality of neural network for classification of weather forecasting is achieved, the training would be a one-time affair, and the so trained network could be used for weather forecasting. The performance of the ENN is compared with the performance of Functional Graph based ENN for weather forecasting. Also the performance of ONN is compared with the performance of Functional graph based opto-electronic neural network for weather forecasting. In ENN, the training is terminated in eight thousand two hundred and thirty six iterations [17]. In the Functional Graph based ENN, the training is terminated in three thousand nine hundred and eighty eight iterations. The training of Functional Graph based ENN is fast compared to ENN. It is represented in Fig.4. The accuracy of Functional Graph based ENN (FGENN) is as good as ENN. In ONN, the training is terminated in one thousand one hundred and sixty iterations. In the Functional graph based opto-electronic neural network the training is terminated in nine hundred and seventy seven iterations. The training of Functional graph based opto-electronic neural network is fast compared to ONN. It is also represented in Fig.4. The accuracy of Functional Graph based Opto-electronic Neural Network (FGONN) is as good as ONN. The Functional graph based ENN and ONN models solve X-OR problem.

Table 1 Performance Classification

Weather Forecasting	No. of Patterns	Percentage of correct Classification					
		Expert I	Expert II	ENN	Functional Graph ENN	ONN	Functional Graph ONN
No rain	12	92	75	92	92	75	75
Moderate rain	12	83	58	58	67	67	67
Heavy rain	12	75	92	92	92	92	92
Accuracy %		83	75	81	84	78	78

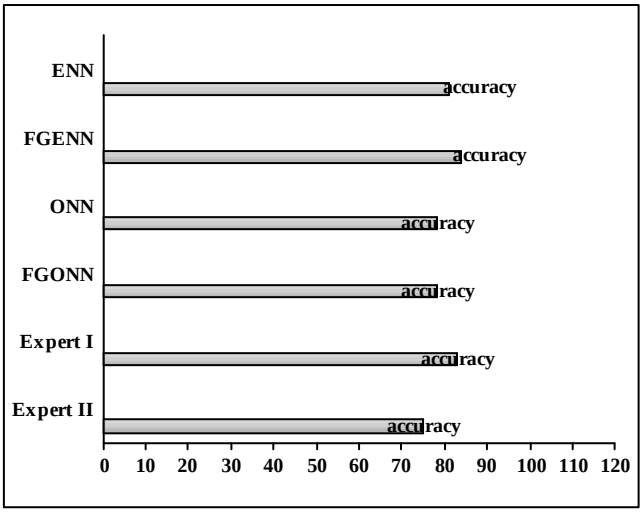

Fig.3. Over all Performance Classification
ENN, FGENN, ONN, FGONN, Expert I, Expert II

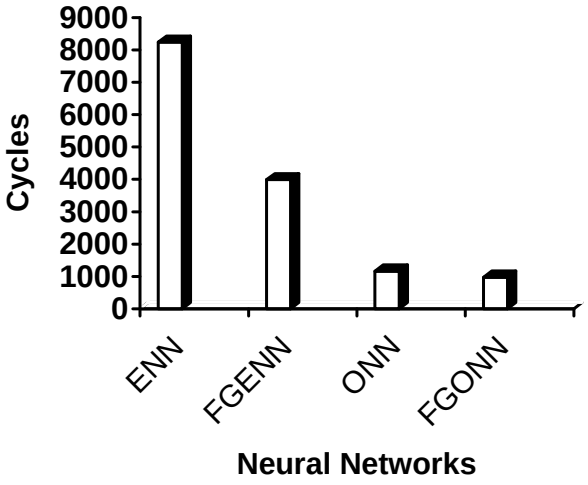

Fig.4 Performance of Neural Networks

8 Conclusion

The weather forecasting has been carried out successfully using ENN, Functional Graph based ENN, Opto-electronic neural network and Functional graph based opto-electronic neural network. The accuracy of weather forecasting, obtained using neural network models is compared with two meteorological experts and reported. The performance of Functional Graph based ENN, which is better than the performance of ENN, is reported. Also the performance of Functional graph based opto-electronic neural network, which is better than the performance of Opto-electronic neural network, is reported. The functional graph based neural network models solve X-OR problem. This study will encourage researchers to use Functional Graph based neural network models for weather forecasting. Efforts are in progress to reduce the time consumed by back propagation training algorithm of neural networks. Dynamic inclusion of new factors may be incorporated to improve its adaptability.

Acknowledgement

The author acknowledges with thanks all necessary support extended by the Rector of European University of Lefke for successful completion of the research paper.

References

[1] W. Dabberdt, Weather for Outdoorsmen: A complete guide to understanding and predicting weather in mountains and valleys, on the water, and in the woods, Scribner, New York, 1981.

[2] Christopher M. Bishop, Neural networks for pattern Recognition, Oxford University Press, New Delhi, 1995.

[3] Sheng Ma and Chuanyi Ji, Performance and Efficiency: Recent advances in Supervised Learning, Proceedings of the IEEE, vol. 87, no. 9, September 1999.

[4] -----, Universal approximation bounds for artificial neural networks, IEEE Trans. Inform. Theory, vol.39, pp. 930-944, May 1993.

[5] Limin Fu, Neural Networks in Computer Intelligence, Tata McGraw hill, 2003.

[6] Ra´ul Rojas, Neural Networks-A Systematic Introduction, Springer-Verlag, Berlin, 1996.

[7] V. Joseph Raj, et.al., Performance Analysis of Wireless Network using Opto-Electronic Neural Network, Karpagam Journal of Computer Science, Volume 2, Number 3, 2006, pp. 191-195.

[8] Martin Heusse, Franck Roussean, Romaric Guileer, Andrzej Duda, Idle sense: An optical access method for High Throughput and Fairness in Rate Diverse Wireless LANs, Computer Communication Review, Volume 35, Number 4, October 2005.

[9] Yanjing Sum, et.al., Construction of Virtual Backbone on Growth-Bounded Graph with Variable Transmission Range, WSEAS TRANSACTIONS on COMPUTERS, vol.7, no.1, January 2008, pp.32-38.

[10] Shih-Jung Wu, et.al., Faulty-Tolerant Algorithm for Mapping a Complete Binary Tree in an IEH, WSEAS TRANSACTIONS on COMPUTERS, vol.7, no.3, March 2008, pp.83-88.

[11] Wesam Barbakh, et.al., A Novel Construction of Connectivity Graphs for Clustering and Visualization, WSEAS TRANSACTIONS on COMPUTERS, vol.7, no.5, May 2008, pp.424-434.

[12] Larbi Sekhri, et.al., Diagnosability of Automated Production Systems Using Petri Net Based Models,

IEEE International Conference on Systems, Man and Cybernetics, 2004, pp.5091-5096.
[13] A. Ghariani, et.al., A Functional Graph Approach for Alam Filtering and Fault Recovery for Automated Production Systems, Proceedings of the Sixth International Workshop on Discrete Event Systems (WODES), IEEE, COMPUTER SOCIETY, 2002, pp.1-6.
[14] Igor T. Podolak , Functional Graph Model of a Neural Network , IEEE Transaction on systems, amn, and cybernetics – part B : cybernetics, Vol. 28, No. 6, 1998.
[15] V. Joseph Raj, et.al., Orthonormal Implementation on Neural Network – A New Approach, International Conference on Artificial Intelligence (ICAI), WORLDCOMP, LasVegas, USA, 2007, Vol.I, pp.221-224.
[16] V. Joseph Raj, et.al., Orthonormal Implementation on Opto-Electronic Neural Network – A New Approach, Second International Conference on Internet Technologies and Applications (ITA), North Wales, UK, 2007, pp. 489-494.
[17] V. Joseph Raj, et.al., Data Mining Based Neural Network Approach for Weather Forecasting, Second International Conference on Emerging Adaptive Systems and Technologoies – NICE EAST, India, 2007, pp. 458-460.

Dr.V. Joseph Raj is an Associate Professor in the Department of Computer Engineering, European University of Lefke, TRNC, Turkey. He obtained his post graduate education from Anna University, Madras, India and his PhD degree in Computer Science from Manonmaniam Sundaranar University, Tirunelveli, India. He is guiding Ph.D. scholars of various Indian Universities. His research interests include artificial neural network, image processing, networks, and operations research. He has great flair for teaching and has twenty two years of teaching experience. He has published 40 research papers at International and National levels.