Journal of Computer Science 8 (1): 18-25, 2012
ISSN 1549-3636

Cerebrovascular Accident Attack Classification Using Multilayer Feed Forward Artificial Neural Network with Back Propagation Error

[1]Olatubosun Olabode and [2]Bola Titilayo Olabode
[1]Department of Computer Science,
Federal Universityof Technology, Akure, Nigeria
[2]Department of Mathematical Sciences,
Olabodebola Federal University of Technology, Akure, Nigeria

Abstract: Problem statement: Most important problems of medical diagnosis. When there is a cerebrovascular accident attach the chances of a successful treatment depends essentially on the early diagnosis. In practice the part of medical errors while diagnosing a stroke type comes to 20-45% even for experienced doctors and the scope of methods of neurovisualization at stroke diagnosis are limited. **Approach:** In this research study, attempt was made to model the application of Artificial Neural Networks to the classification of patient Cerebrovascular Accident Attack. The Network for the consisted of a three-layer feed forward artificial neural network with back-propagation error method. **Results:** Data were collected from 100 records of patients at Federal Medical Centre Owo, Nigeria and the Artificial Neural Networks classifier was trained using gradient decent backward propagation algorithm with flexible sigmoid activation function at one hidden layer, with 16 inputs nodes representing stroke onset symptoms at the input layer, 10 nodes at the hidden layer and one node at the output layer representing the type of the attack. **Conclusion:** The learning Rate γ was set between 0.1 and 0.9 while the epoch set at 150. Initial weight set at Rand (-0.5 and 0.5). The simulation results showed that the model was capable of producing a reasonable forecasting accuracy in short.

Key words: Sigmoid function, back-propagation, gradient descent, cerebrovascular accident, ischemic stroke, hemorrhagic stroke, Myocardial Infarction (MI), Computed Tomography (CT), Mean Square Error (MSE), artificial neural network

INTRODUCTION

The word stroke (Cerebrovascular accident attack) is used to refer to a clinical syndrome, of presumed vascular origin, typified by rapidly developing signs of focal or global disturbance of cerebral functions lasting more than 24 h. or leading to death.

Usually, when there is an attack, the chances of a successful treatment depend essentially on the early diagnosis. In the early stage, many cases of stroke are curable if properly managed. In case of false or wrong diagnosis, precious times are lost, waste of scarce health and social services resources, the chances of curing/recovering from the attack diminishes and may eventually lead to death of patient. Stroke which has been categorized Mosalov *et al*., (2007) as ischemic stroke, hemorrhagic stroke and subarachnoid hemorrhage and has varying medical treatment in each case. In practice the part of medical errors while diagnosing a stroke type comes to 20-45% even for experienced doctors. According to the discussion in Jehangir and Rehman (2005), differentiation of cerebral infarction and cerebral hemorrhage is the most important first step in the management of acute stroke as clinical management of the two disorders differs substantially. In most developed countries, diagnosis is easily obtained by CT scanning, which allows the accurate distinction of hemorrhagic and ischemic types. However, quick access to CT scanning is not available in every country and hospitals. It is well known that some clinical data may suggest a hemorrhagic or ischemic stroke even though no data are specific enough to allow a reliable diagnosis.

A number of scoring systems based on clinical data determining the relative likelihood of infarction or hemorrhage were developed and tested over the last decade. Although the clinical diagnoses made using these scores seem more accurate than those made by physicians, they present several problems.

Corresponding Author: Olatubosun Olabode, Computer Science Department, Federal University of Technology, Akure, Nigeria

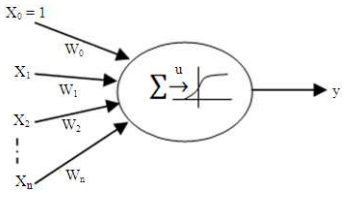

Fig. 1: Artificial model of neuron

The scope of methods of neurovisualization at stroke diagnosis is limited. Thus, the development of intellectual computer support systems that could assist to decrease the part of medical errors. In this study, attempt is made at looking at the application of Neural Networks for classification of Cerebrovascular Accident Attack (stroke) in patient. A Neural Network has the ability to mimic this type of decision-making process and use a knowledge base of information and a training set of practice cases, to learn to diagnose diseases.

The behavior of an artificial neural is inspired by the assumed behavior of a real neural in organic networks (Veelenturf, 1995). A simplified model of real neuron is composed of a cell body or soma, a set of fibers entering the cell body, called the dendrites and one special fiber leaving the soma, called the axon. The simplified model of a neuron can be simulated by an artificial neuron, Fig. 1. (Krose and Smagt, 1996; Mitchell *et al*., 1990.

The perceptron is the basic processing element. It has inputs that may come from the environment or may be the outputs of other perceptrons.

Associated with each input, $x_j \in R$,I = 1,2,…n, is a connection weight or synaptic weight $w_j \in R$ and the output, y, in the simplest case is a weighted sum of the inputs Eq. 1:

$$y = \sum_{i=1}^{n} w_i x_j(t) + w_o \quad (1)$$

w_o is the intercept value to make the model more general; it is generally modeled as the weight coming from an extra bias unit, x, which is always +1.

MATERIALS AND METHODS

The Multi-layer perceptrons artificial neural networks with back-propagation error method are feed-forward nets with one or more layers of nodes between the input and output nodes. These additional layer contains hidden units or nodes that are not directly connected to both the input and output nodes. Multilayer perceptron with back-propagation learning is perhaps the most common paradigm for supervised neural network computing to date. This has been observed in the medical imaging area as well as many other pattern recognition areas.

ANNs can be viewed as weighted directed graphs in which artificial neurons are nodes and directed edges with weights are connections between neuron outputs and neuron inputs (Jain *et al*., 1996). Artificial network consists of the following as discussed in (Rumelhart and McClelland, 1986):

Processing units: Within neural systems it is useful to distinguish three types of units: input units (indicated by an index i) which receive data from outside the neural network, output units (indicated by an index o) which send data out of the neural network and hidden units (indicated by an index h) whose input and output signals remain within the neural network, (Ethem, 2004; Mesut and Bob, 2004). During operation, units can be updated either synchronously or asynchronously. (Krose and Smagt, 1996; Haykin, 1999; Jones, 2008).

Connections between units: In most cases we assume that each unit provides an additive contribution to the input of the unit with which it is connected. The total input to unit k is simply the weighted sum of the separate out puts from each of the connected units plus a bias or offset term θ_k Eq. 2:

$$s_k(t) = \sum_{j}^{l} w_{jk}(t) + \theta_k(k) \quad (2)$$

The contribution for positive wjk is considered as an excitation and for negative wjk as inhibition. In some cases more complex rules for combining inputs are used, in which a distinction is made between excitatory and inhibitory inputs.

Activation and output rules: We also need a rule which gives the effect of the total input on the activation of the unit. We need a function fk which takes the total input sk (t) and the current activation yk and produces a new value of the activation of the unit k Eq. 3:

$$y_k(t+1) = f_k(y_k(t), s_k(t)) \quad (3)$$

Often, the activation function is a non-decreasing function of the total input of the unit Eq. 4:

$$y_k(t+1) = f_k(y_k(t)) = f_k(\sum_{j}^{l} w_{jk}(t)_{yj} + \theta_k(t) \quad (4)$$

Although activation functions are not restricted to non-decreasing functions. Generally, some sort of threshold function is used. This includes a hard limiting threshold function (a sign function), or a linear or semi-linear function, or a smoothly limiting threshold (Joarder *et al*., 2006; Tom, 1997)

Medical Doctors use a combination of a patient's case history and current symptoms to reach a health diagnosis when a patient is ill. In order to recognize the combination of symptoms and history that points to a particular disease, the doctor's brain accesses memory of previous patients, as well as information that has been learned from books or other doctors. One of the most important problems of medical diagnosis is the subjectivity of the specialist, in particular pattern recognition activities, that is, the experience of the professional is closely related to the final diagnosis. This is due to the fact that the result does not depend on a systematized solution but on the interpretation of the patient's signal. It was highlighted in Brause (2001) that almost all the physicians are confronted during their formation by the task of learning to diagnose. Where, they have to solve the problem of deducing certain diseases or formulating a treatment based on more or less specified observations and knowledge.

An ANN is an artificial intelligence tool that can identifies arbitrary nonlinear multiparametric discriminate functions directly from clinical data. The use of ANNs has gain increasing popularity for applications where description of the dependency between dependent and independent variables is either unknown or very complex. The application of ANNs to complex relationships makes them highly attractive for the study of complex medical decision making. Recent applications include (Mesut and Bob, 2004) diagnosis, staging and progression of prostate cancer, progression of benign prostate hyperplasia and bladder cancer recurrence in TA/T1 bladder cancers.

RESULTS AND DISCUSSION

Predictive models are used in a variety of medical domains for diagnosis and prognostic tasks. These models are built from "experience", which constitutes data acquired from actual cases. The data can be preprocessed and expressed in a set of rules, such as it is often the case in knowledge based expert system, or serve as training data for statistical and medical learning models. Among the options in the latter category, the most popular models in medicine are logistic regression and artificial neural networks, (Mesut and Bob, 2004). ANNs are nonlinear regression computational devices that have been used for over 45 years in classification and survival prediction in several biomedical systems, including colon cancer. Multilayer feed-forward neural networks have been widely used for financial forecasting due to their ability to correctly classify and predict the dependent variable (Vellido *et al*., 1999). Back propagation is by far the most popular neural network training algorithm that has been used to perform learning for multilayer feed-forward neural networks (Russell and Norving, 1995).

Ay *et al*. (2010) there is currently no instrument to stratify patients presenting with ischemic stroke according to early risk of recurrent stroke. However, in the Communications and Public Liaison, (2008) the symptoms of stroke are distinct because they happen quickly, for instance Sudden numbness or weakness of the face, arm, or leg (especially on one side of the body), Sudden confusion, trouble speaking or understanding speech, Sudden trouble seeing in one or both eyes, Sudden trouble walking, dizziness, loss of balance or coordination and Sudden severe headache with no known cause. So it was suggested that the best treatment for stroke is prevention. There are several risk factors that increase a chance of having a stroke, summarily, this includes, High blood pressure, Heart disease, Smoking, Diabetes, High cholesterol.

Cerebrovascular Accident is an acute focal neurological deficit resulting from cerebrovascular disease. Notes, It is difficult to be sure clinically about the type of stroke because in majority of cases as there are no specific differentiating feature. However, findings has shown that features like sudden onset of coma or changing state of consciousness with severe headache, vomiting and meningeal irritation could suggest intracranial bleed.

Similarly in cerebral infarction patient are usually presented with sudden onset of stroke with lateralizing neurological deficit (hemi paresis, aphasia, homonymous hemianopia) with or without clinically detectable risk factors such as hypertension, atrial fibrillation, rheumatic heart disease, recent myocardial infarction, (Guieis and Perez, 1988).

Algorithm for Network Training: A feed-forward network has a layered structure. Each layer consists of units which receive their input from units from a layer directly below and send their output to units in a layer directly above the unit. There are no connections within a layer. The Ni inputs are fed into the rest layer of Nh,1 hidden units. The input units are merely 'fan-out' units; no processing takes place in these units. The activation of a hidden unit is a function F1 of the weighted inputs plus a bias.

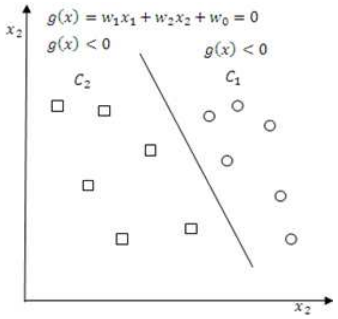

Fig. 2: A case of two dimensional linear discriminate as divided by a line

The output of the hidden units is distributed over the next layer of Nh, 2 hidden units, until the last layer of hidden units, of which the outputs are fed into a layer of No output units. Although back-propagation can be applied to networks with any number of layers. But, In most applications a feed-forward network with a single Layer of hidden units is used with a sigmoid activation function for the units (Hornik *et al*., 1989; Funahashi, 1989; Cybenko, 1989; Hartman *et al*., 1990).

For a general three-layer network with n1 neurons in the first and n2 neurons in the second layer and n_3 in the third layer as presented in Fig. 2.

Gradient descent adaptation method: For any output neurons we can derive the relation between the output vector y and the input vector x= $[x_{1,}x_2,\ldots,x_{no}]$.

For the value of the ith output in the third layer we have Eq. 5:

$$g_i(x) = \max_{j=1}^{k} g_j^{(x)} \tag{5}$$

With f θ_{3i} the transfer function of the particular output neuron and 3i the weighted input of that neuron defined by Eq. 6:

$$\begin{aligned} g(x) &= g_1(x) - g_2(x) \\ &= (w_1 + w_2)^T x - (w_{10}^T + w_{20}) \\ &= (w_1^T + w_{10}) - (w_2^T + w_{20})(w^T x + w_0) \end{aligned} \tag{6}$$

With w3t.j the weight in the connection between the j th neuron in the third layer and the output Y_{2J} of neuron J in the second layer and W3t.0 the threshold weight of the particular output neuron.

For the value of the j th output y_{2j} in the second layer we obtain similarly Eq. 7 and 8:

$$\begin{cases} C_1 & \text{ifs}(w^T x > 0 \\ C_2 & \text{otherwise} \end{cases} \tag{7}$$

With

$$y_{3j} = f_{3j}(s_{3j}) \tag{8}$$

For the value of the kth output y_1k in the first later we have Eq. 9:

$$s_{3j} = \sum_j w_{3j} y_{2j} + w_{3j,0} \tag{9}$$

For the value of the weighted input of the kth neuron in the first layer we have Eq. 10:

$$y_{2j} = f_{2j}(s_{2j}) \tag{10}$$

With x_m the mth input value.

Definition: A simple perceptron is a computing unit with threshold θ which, when receiving the n real inputs $x_1,x_2,\ldots,x_n$ through edges with the associated weights $w_1, w_2,\ldots,w_n$ outputs 1 if the inequality $\sum_j^n = 1 w_j x_j \geq \theta$ holds and otherwise 0 (Raul, 1996).

By substitution we obtain from the equations 10 above for the relation between the ^{j} th output and the inputs: $x_1,x_2,\ldots,x_m$ of the network Eq. 11:

$$\begin{aligned} y_{3j} = f_{3t}[\sum_j w_{3j,j} f_{2j}\{\sum_j w_{2j,k} f_{1k}(\sum_j w_{1km} x_m \\ + w_{1k,0}) + w_{2j,0}\} + w_{3j,0}] \end{aligned} \tag{11}$$

With the summation over the number of neurons in the particular layer.

By introducing a dummy neuron in layer 2 and 1 with constant transfer functions f 20(.) = 1 f 10 (.) = 1 and by adding a constant additional input x0 = 1 to the net, we can incorporate the thresholds $w_{pq,0}$ of the neurons in the weighted input of the neurons by the terms $w_{3t},0f_{20}(.),w_{2j},f_{10}$ (.) and $w_{1k,0}x_c$ (Veelenturf, 1995).

Theorem 1: A function g_{w1} $R_n \rightarrow R$ that can be realized by a sigle-neuron perceptron with transfer function f can go exactly through the samples of the data set $[x_{j,}$ t $(x_j)] \in D$, iff for every element $[x_j,t(x_j)]$ from D the vector, Eq. 12:

$$\tilde{x}_j = \{\tilde{x}_j, ^{f-1}[t(x_j)]\} \text{ with } \tilde{x}_j = [1, x_{j1}, x_{j2}, \ldots, x_{jn} \tag{12}$$

Is a linear combination of some unique set of n+ 1 linear independent vector?

$$\tilde{x}_j = \{\tilde{x}_j, {}^{f-1}[T(x_j)]\}$$

Obtained from n+1 elements $[x_j,t(x_j)]$ from the data set D.

The sigmoid function is the commonly used transfer function (Russell and Norving, 1995). It is given as Eq. 13-15:

$$f[s(xj)] = \frac{1}{1+e^{-s(xj)}} \quad (13)$$

And:

$$s(x_j) = \sum_{j=0}^{n} w_j * x_{ij} \quad (14)$$

Then:

$$\frac{df}{ds} = f(s).[1-f(s)] \quad (15)$$

Activation can take any real value, including negative values, if the particular output activation function supports this. In Jain *et al*. (1996), the supervise learning paradigm, the network is given a desired output for each input pattern. During the learning process, the actual output y generated by the network may not be equal to the desired output d. The basic principle of error-correction learning rule is to use the error signal/function to modify the connection weight to gradually reduce this error.

Theorem 2: Given a single-neuron perceptron that can realize functions $g_w:R^n \rightarrow R$, then the weight vector w of the function $g_w:R^n \rightarrow R$,with a minimal MSE for a given data set ^{D} can be determined from a set of linear equations if:

- The 'error' $\Delta g_w(x_j) = g_w(x_j) - t(x_j)$ for all x_i of D is 'small'
- The derivative dE⁄ds = 0 only for its minimum
- The transfer function f of the neuron has an inverse f^{-1}

Theorem3: The MSE of a multi-layer network will decrease if a weight w_1 is changed according to $\Delta w_j = -\varepsilon.\frac{\partial E}{\partial_{wj}}$ for some ε >0 sufficiently small.

Theorem 4: The MSE of a multi-layer network will decrease if a weight w is changed according to:

$$\Delta w = \varepsilon.\nabla E(w)$$

For some ε >0 sufficiently small with ∇E(w) the gradient vector with respect to w. The proportionality constant ε is called the learning rate. However the learning rate as prescribed in the above theorem is frequently called the learning rule based on the gradient descent method of the MSE (Veelenturf, 1995).

Given some finite learning set $L \subseteq D$ of N pairs $x_j,t(x_j)]$. Let u be the set of input vectors in ^{L}. We want to have for each input vector $x_j \in u$ some target output vector $t(x_j)$. Let $tj(x_j)$ be the target value of the j th output neurons for input vector x and let $yj(x_i)$ be the actual output of the j th neuron in the output layer of the neural network. Then the sum square error from using the Adaline on all training patterns is given by Eq. 16:

$$E = \frac{1}{N}\sum_{x_i \in U} n(x_i)\sum_{j=1}^{n_3}[t_j(x_i) - y_i(x_i)]^2 \quad (16)$$

With N the number of samples in the learning set L, n (x_j), the number of times input x occurs in u And n_3 the number of output neurons in the third layer, (Veelenturf, 1995) Eq. 17:

$$E = \frac{1}{2}\sum_{x_i \in U}\sum_{j=1}^{n_3}[t_j(x_i) - y_j(x_i)]^2 \quad (17)$$

Equation 20 is called energy or error function of the net. The error E is a function of all weights in the net. For a given finite learning set L ans some initial random distribution of the weights the MSE will in general be large. We want to change, during a so-called learning period, the weights step by step, such that E wills degease to its global minimum. Only when the structure of the neural net is able to realize exactly the learning set L must the minimum of E become zero? For an infinitesimal increment Δw_j of the weight w_j in the connection of some neuron, somewhere in the net the following holds Eq. 18:

$$\Delta E = \frac{\partial E}{\partial w_i}.\Delta w_i \quad (18)$$

Weight adjustment variants: With back propagation, there are a number of other weight adjustment strategies that can be applied that can speed learning or avoid becoming trapped in local minima. One of such involves a momentum term where a portion of the last weight change is applied to the current weight adjustment round. The following equation shows an updated weight adjustment variant based on Eq. 19:

$$w_{ij} = w_{ij} + \varepsilon E u_j \quad (19)$$

For the given Error (E) and activation (or cell output, U_1), we multiply by a learning rate $\in$ and add this to the current weight. The result is a minimization of the error at this cell, while moving the output cell activation closer to the expected output. The difference is that a portion of the last weight change (identified as Δw_{ij} is accumulated using a small momentum multiplier (m) Eq. 20:

$$w_{ij} = w_{ij} + \varepsilon E_j + m\Delta w_{ij} \quad (20)$$

The last weight change is stored using Eq. 21:

$$\Delta w_{ij} = \varepsilon E u_i \quad (21)$$

The adaptation of weight w_j could be obtained using Theorem 1, that is Eq. 22:

$$\Delta w_j = -\varepsilon \frac{\partial E}{\partial w_j} \quad (22)$$

According to theorem 2, the learning implies for the adaptation of the weights connecting neurons j in the second layer to neurons i in the third layer Eq. 23:

$$\Delta w_{3i,j} = -\varepsilon \frac{\partial E}{\partial w_{3j}} = \varepsilon \sum_{x_i \in U} [t_{3j}(x_j) - y_{3i}(x_i)] \frac{df_{3i}}{ds_{3i}} \quad (23)$$

The adaptation of the weights connecting neuron k in the first player to neuron j in the second layer Eq. 24:

$$\Delta w_{2j.k} = -\varepsilon \frac{\partial E}{\partial w 2 j.k} = \varepsilon \sum_{x_i \in U} \{t_3(x_i) - y_3(x_i)\} \frac{df_3}{ds_3} y_{2k}(x_i) \quad (24)$$

If we represent all the output of the n_1 neurons in the first layer including the constant component $x_0 = 1$ with a vector $\hat{x}$, then we can write, with the use of the extended weight vector $\hat{w}_{2j} = [w_{2j,0}.w_{2j.0},...,w_{2j.0,}]$ for the adaptation of the weight vector w_{2j} of output neuron j Eq. 25:

$$\Delta w_{2j} = \varepsilon \sum_{x_i \in U} \{t_{2j}(x_i) - y_{2j}(x_i)\} \frac{df_{3i}}{ds_{3i}} \hat{y}_{2k} \quad (25)$$

The adaptation of the weights connecting neuron input to the first layer Eq. 26:

$$\Delta w_{1j.m} = -\varepsilon \frac{\partial E}{\partial w_{1i.m}} = \varepsilon \sum_{x_i \in U} \{t_3(x_i) - y_3(x_i)\} \frac{df_3}{ds_3} w_{2k} \frac{df_1}{ds_1} x_{1i} \quad (26)$$

With the notation $\delta_{2j}(x_i) = t_{2j}(x_i) - y_{2i}(x_i)$ for the error of the output neuron t for an input x_i we can thus write the following theorem (Veelenturf, 1996).

Theorem 5: The adaption of the weight vector w of a neuron in the second layer of a two-layer perceptron, in order to minimize the MSE for a given set of target values:

$$\Delta w_{2j} = \varepsilon \sum_{x_i \in U} \delta_{2j}(x_i)\} \frac{df_{2j}}{ds_{2j}} \hat{y}_1$$

For the adaption of the weight connecting the net input with neuron k in the first layer, we obtain according to Theorem 2 Eq. 27:

$$\Delta w_{1k,m} = \varepsilon \sum_{x_i \in U} \sum_{j=1}^{n_2} \{ t_{2j}(x_i) - y_{2j}(x_i)\} \frac{df_{2j}}{ds_{2j}} w_{2j.k} \frac{df_{1x}}{ds_{1k}} x_m \quad (27)$$

The sum of product over j can be considered as the error (from the output neurons back-propagated) of neuron k in the first layer. So with Eq. 28:

$$\delta_{1k}(x_i) = \sum_{j=1}^{n_2} \{t_{2j}(x_i) - y_{2j}(x_i)\} \frac{df_{2j}}{ds_{2j}} w_{2j.k} \quad (28)$$

We can write Eq. 29:

$$\Delta w_{1k,m} = \varepsilon \sum_{x_i \in U} \delta_{1k}(x_i) \frac{df_{1k}}{ds_{1k}} x_m \quad (29)$$

If we represent all the input of the network, including the constant component, with a vector, then we can write, with the use of the extended weight vector, the following theorem for the adaptation of the weight vector of input neuron k:

Theorem 6: The adaptation of the weight vector of a neuron in the first layer of a two-layer perception in order to minimize the MSE for a given finite set U of input vectors, is:

$$\Delta \hat{w}_{ik} = \varepsilon \sum_{x \in u} \delta_{1k}(x_i) \frac{df_{1k}}{ds_{1k}} \hat{x}_i$$

We define the internal learning rate for input vector for the weights of neuron k in the layer as:

$$y_{1k}(x_i) = \delta_{1k}(x_i) \frac{df_{1k}}{ds_{1k}}$$

With this definition we obtain for the adaptation of the weights of neuron k in the first layer due to input:

$$\Delta w_{1k} = \varepsilon \sum_{x \in u} \gamma_{1k}(x_i)\hat{x}_i$$

The sum of weighted by the scalars - extended input vectors will be denoted by:

$$\hat{x} = \varepsilon \sum_{x \in u} \gamma_{1k}(x_i).\hat{x}_i$$

Thus we can write for the adaptation:

$$\Delta \hat{w}_{1k} = \varepsilon.\hat{x}$$

We see that after adaptation the weight vector of neuron k in the first layer will turn in the direction of the weighted sum of input vectors. This implies that the separating hyper plane in the extended input space, as well as the hyper plane w.x = 0 in the non-extended input space, of neuron k will turn in a direction 'perpendicular' to the weighted sum of input vectors.

Data source: Data for this research work was collected from Federal Medical Centre, Owo, Nigeria. Records of patients suffering from cerebrovascular accident attack were considered. Values obtained from the records forms the input variables in the input layer with 16 nodes which includes:

X_1 = Age
X_2 = Sex
X_3 = Hypertension
X_4 = Rate of onset {Sudden, Gradual}
X_5 = Activity: During activity { }
X_6 = Activity: At rest { }
X_7 = Vomiting {Present, Absent}
X_8 = Headache {Present, Absent}
X_9 = Loss of Consciousness {Stupor/Coma, Alert}
X_{10} = Transient ischemic attack {Absent, Present}
X_{11} = Blood Pressure {Moderate/severe HTN, Normal/mild}
X_{12} = Cerebrospinal Fluid {Bloody CSF, Normal}
X_{13} = Siriraj Stroke Score (SSS) { }
X_{14} = Admission pulse pressure
X_{15} = Atrial fibrillation Present, Absent}
X_{16} = Locomotor brachialis{Present,present}
X_{17} = Type of stroke {Ischemic, hemorrhagic}

The hidden layer of the network consists of 10 nodes. A good method to find out how many hidden neurons are needed would be to train the network several times (10+) and see what the average accuracy is at the end and then use this accuracy as a measure to compare different architectures.

Table 1: Validation data

E	TSA	GSA	TSM	GSM
0	6.5750	28.100	0.146539	0.130001
1	30.0583	37.225	0.127413	0.118761
2	36.5333	41.700	0.117833	0.108638
144	64.5167	61.825	0.066149	0.072690
145	64.1667	57.550	0.066278	0.067984
146	65.9083	59.325	0.063510	0.070262
145	66.6750	62.525	0.061974	0.066581
146	65.5000	61.925	0.061092	0.069056
147	65.4417	60.450	0.060699	0.069784
148	65.9167	62.025	0.061336	0.066428
149	65.9417	64.000	0.062089	0.071178
150	65.9417	64.000	0.062089	0.071178

Elapsed Epochs: 150 Validation Set Accuracy: 64.45 Validation Set MSE: 0.0698843 Key: E-epoch; TSA-Training set; GSA-Generalized set accuracy; TSM-Training set MSE; GSM-Generalized set MSE

The weights between the layers are randomly initialized. They are set to small values in the range of [-0.5, 0.5]. Technically the weights can be set to a any fixed value since the weight updates will correct the weights eventually but may be inefficient. The whole point of setting the initialization range is to reduce the number of epochs required for training. Using weights closer to the required ones will obviously need fewer changes than weights that differ greatly. The learning Rate θ was set between 0.1 and 0.9 while the epoch set at 10.

Classically we split the data set into three parts: (The classic split of the dataset is 60, 20 and 20% for the training, generalization and validation data sets respectively):

- The training data is what we used to train the network and update the weights with so it must be the largest chunk
- Generalization data is data that we will run through the NN at the end of each epoch to see how well the network manages to handle unseen data
- The validation data will be run though the neural network once training has completed (Table 1)
- The deciding rule to stop training is to stop once a set number of epochs have elapsed.

CONCLUSION

In this research study, the application of Artificial Neural Networks to classification of Cerebrovascular Accident Attack in a patient is considered. The described is the theory behind the classification of Cerebrovascular Accident Attack in a patient using the three-layer feed forward artificial neural networks with back-propagation error. Data were collected for 100 records (60 males and 40 females) of patients from Federal Medical Centre, Owo, Nigeria and the Artificial Neural Networks classifier was trained using backward propagation algorithm with flexible sigmoid activation function at one hidden layer with 16 inputs nodes

representing stroke onset symptoms at the input layer, 10 nodes at the hidden layer and one node at the output layer representing the type of attack. The learning Rate y was set between 0.1 and 0.9 while the epoch set at 10. The network was successfully trained with 150 epoch and MSE of 0.0698843.

REFERENCES

Ay, H., L. Gungor, E.M. Arsava, J. Rosand and M. Vangel *et al*., 2010. A score to predict early risk of recurrence after ischemic stroke. Neurology, 74: 128-135. DOI: 10.1212/WNL.0b013e3181ca9cff

Brause, R.W., 2001. Medical analysis and diagnosis by neural networks. Lec. Notes Compu. Sci., 2199: 1-13, DOI: 10.1007/3-540-45497-7_1.

Cybenko, G., 1989. Approximation by superposition of a sigmoidal function. Math. Control, Signals Syst., 2: 303-314. DOI: 10.1007/BF02551274.

Ethem, A., 2004. Introduction to Learning Machine. 1st Edn. MIT Press, Cambridge, Mass, pp: 415, ISBN-10: 0-262-01211-1

Funahashi, K., 1989. On the approximate realization of continuous mappings by neural networks. Neural Networks, 2: 183-192. DOI: 10.1016/0893-6080(89)90003-8

Hartman, E.J., J.D., Keeler and J.M. Kowalski, 1990. Layered neural networks with Gaussian hidden units as universal approximations. Neural Computation 2: 210-215. DOI:10.1162/neco.1990.2.2.210.

Haykin, S., 1999. Neural Networks: A Comprehensive Foundation. 2nd Edn. Englewood Cliffs, N.J, Prentice-Hall Inc,. pp: 842. ISBN: 0132733501 9780132733502

Hornik, K., M. Stinchcombe and H. White, 1989. Multilayer feedforward networks are universal approximators. Neural Networks, 2: 359-366. DOI: 10.1016/0893-6080(89)90020-8.

Jain, A.K. and J. Mao and K.M. Mohiuddin 1996. Artificial neural networks: a tutorial. IEEE Comput., Sci. Eng., 29: 31-44. DOI: 10.1109/2.485891

Jehangir, K. and A. Rehman, 2005. Comparison of clinical diagnosis with computed tomography in ascertaining type of stroke. J. Ayub. Med. Coll. Abbottabad, 17: 65-75.

Joarder, K., R. Begg, R.A. Sarker, 2006. Artificial Neural Networks in Finance and Manufacturing. 1st Edn. Hershey, PA: Idea Group, united state, pp: 299.

Jones, M.T., 2008. Artificial intelligence a systems approach, neural network. 1st Edn. Sudbury, Mass, Jones and Bartlett Publishers, England, pp: 498, ISBN: 9780763773373

Jones, T.M., 2008. Artificial Intelligence A Systems Approach, Neural Network.1st Edn. Jones and Bartlett Learning, Burlington. pp: 498,

Krose, B. and P.V.D. Smagt, 1996. An Introduction to Neural Networks. 8th Edn.

Mesut, R. and B. Djavan, 2004. Artificial neural networks in urology. Eur. Urol. Supplements, 3: 1-88.

Mitchell, T., B. Buchanan, G. DeJong and T. Dietterich and P. Rosenbloom *et al*., 1990. Machine learning; annual review of computer science. J., Comput. Sci., 4: 417-433 DOI: 10.1146/annurev.cs.04.060190.002221

Mosalov, O.P., O.Y. Rebrova and V.G. Red'ko, 2007. Neuroevolutionary method of stroke diagnosis. Optical Memory Neural Networks (Information Optics). 16: 99-103. DOI: 10.3103/S106 0992X07020063

Raul, R., 1996. Neural Networks: A Systematic Introduction. 1st Edn. Spriger. Springer-Verlag Berlin, New York ISBN 3540605053, pp: 502.

Rumelhart, D.E. and J.L. McClelland, 1986. Parallel distributed processing: Explorations in the microstructure of cognition. Cambridge, MA: MIT Press. pp: 611. ISBN:0262631105, 9780262631105

Russell, S.J. and P. Norvig, 1995. Artificial Intelligence: A Modern Approach. 3rd Edn., Pretience Hall, Pearson, France. ISBN-10: 0132126842. ISBN-13: 9780132126847

Tom M.M, 1997. Machine Learning. 1st Edn. McGraw-Hill, New York, ISBN 0070428077, pp: 11-20.

Veelenturf, L.P.J., 1995. Analysis and Applications of Artificial Neural Networks. 1st Edn. Prentice Hall. London and New York, ISBN 013489832X, pp: 259

Vellido, A., P.J.G. Lisboa and J. Vaughan, 1999. Neural networks in business: a survey of applications Expert Systems with Applications. 17:51-70.