International Journal of Computer Science & Communication (ISSN: 0973-7391)
Volume 17 • Issue 2 pp. 557-575 March 2026-Sept 2026 www.csjournals.com

Performance Evaluation of Monolithic Neural Networks for classification of Uterine Cervix Cancer Cases

Mehboob Ali[1], Marziya Arman[2]
[1]Department of Computer Science and IT, University of Ladakh, Kargil, India
[2]Department of Computer Science and IT, University of Jammu, Jammu, India

Abstract:
This experimental study evaluates ten feedforward monolithic neural network algorithms for the classification of uterine cervical cancer into two-class (benign vs. malignant) and seven-class (exact Bethesda system diagnostics) categories. To rigorously train and validate these networks, a novel database of 8,091 individual pap smear cells was constructed, featuring 39 morphological cytoplasm and nuclear parameters per cell extracted via CellProfiler. The dataset comprises 4,956 normal cells and 3,135 abnormal cells. Utilizing MATLAB, a 10-fold cross-validation framework evaluated 19 distinct architectures per algorithm by sweeping hidden nodes from 2 to 20 over 100 epochs. The experimental results revealed that the top three models were the Levenberg-Marquardt Algorithm (12 hidden nodes; 78.6% accuracy), the Variable Learning Rate Gradient Descent Algorithm (18 hidden nodes; 77.3% accuracy), and the One Step Secant Algorithm (16 hidden nodes; 76.2% accuracy), while the Extreme Learning Machine performed the poorest. The study concluded that while the two-class problem is easily solvable, the seven-class problem presents high complexity. Confusion matrix analysis highlighted significant vulnerabilities in Class 2 (normal intermediate) and Class 5 (low-grade lesion) cells across all top models, leading to high false-positive and medically costly false-negative rates. Because standalone monolithic networks failed to deliver sufficient diagnostic reliability, advanced optimization techniques are required to effectively navigate the cervical cancer problem space.

1. Introduction

Cervical cancer occurs when the cervix tissue cells grows and replicates abnormally because of uncontrolled cell division and cell death. This malignant tumor because of uncontrolled cell division forms the cervical cancer. In any type of cancer human body is not able to manage these affected cells for normal usual function rather these cells gets transform into tumor. The tumor if becomes malignant spreads through the blood stream to other parts of the body and

thus infects those parts also. Cervical cancer takes good number of years to develop. The infected cells are called cervical intra-epithelial neoplasia (CIN) also called cervical dysplasia. Therefore any cell on the surface of the cervix if shows any unusual change and may be precursor of cancer are called CIN. Normally in most of the cases the human immune system resolves CIN and do not causes the CIN to show any symptom. But unfortunately in some cases these lesions may develop in to cancer if not treated on time. But this progression to a full cancer from CIN takes many years. Therefore timely and precise diagnosis of cervical cancer is very important. This cancer has become one of the main causes of death among women after breast cancer worldwide and therefore it has become one of the contentious issue of research before the world's scientific community. The main aim of scientific research in cervical cancer aims to have an early diagnostic system so that the cancer may get caught at a very early stage. Among the most common cancer affecting women cervical cancer stands at the fourth place and stands at seventh place among all the common cancers worldwide. As per the reports of WHO death toll of cervical cancer in 2012 was 52800 and about 84% of these cases are reported from poor and developing countries of the world. The main reason of this huge burden of cervical cancer in developing countries is attributed to poor access to medical care and absence of screening and treatment services (Kent A, 2010). Etiology of cervical cancer is still not clear. Medical experts still believe that there is not a single dominant cause of cervical cancer. The only means of avoiding this cancer is early detection and if the cancer is detected early before spreading to other organs the survival rate of patient is more than 97%. Medical research have found that this cancer is caused by a virus called human papillomavirus (HPV) transmitted sexually. About more than 120 types of human papillomavirus (HPV) are reported in medical literature (Chaturvedi Anil et al., 2010). Out of 120 viruses 15 are classified as high risk types (16, 18, 31, 33, 35, 39, 45, 51, 52,) 3 as probably of high risk (26, 53, and 66), and 12 as low-risk (6, 11, 40, 42, 43, 44, 54, 61, 70, 72, 81, and CP6108) (Munoz Nubia et al., 2003). After the HPV virus infects a normal cell the immune system of cell removes the virus but in some cases the virus is not done away by the immune system. The virus then affects the genetic information of the cell causing it to do uncontrolled cell division and growth which causes cervical cancer. There are many factors which causes cervical cancer intercourse and pregnancy at early age, multiple sex partners, having weak immune system, smoking, poor menstrual hygiene etc. normally the early stage of cervical cancer are asymptomatic and with progression of cancer to advanced stages the symptoms starts rapidly

appearing. Some common symptoms of cervical cancer includes bleeding from vagina abnormally, severe pain during vaginal intercourse.

2. Material and Methods

2.1 Cervical Cancer Database

In machine learning it is well known that an algorithm can learn anything and can emulate any intelligent behavior if it is trained on a qualitatively and quantitatively good database. Thus in machine learning the dataset is an important component without which the existence of machine learning is nil. In this experiment with the aim to make the neural network algorithms learn the ground truth of cervical cancer a large database of cervical cancer consisting of 8091 cells of different categories is created. Each cell is represented by a tuple and each cell is characterized by 39 morphological parameters. These morphological parameters corresponds to morphology of cytoplasm and nucleus. Among these 39 features 19 are of cytoplasm feature, 19 are of nucleus features and one feature tells the nucleus to cytoplasm ratio. In addition to these 39 features on more attribute tells the class label of the tuple indicating the class to which the particular cell belongs. In the process of creating this huge database the pathological slides of cervical cancer are collected from Government Medical College, Jammu, Acharaya Shree Chandra College of Medical Science and Hospital, Jammu & Nijjer Pathology Laboratory, Amritsar keeping in care all the medico ethical issues. These pathological slides were analyzed under a multi-headed microscope (NIKON Nikon Eclipse E400 DS-F12). This microscope is attached with a computer and has a mounted camera over it. The camera captures the image of slide under microscope. All the slides were photographed under the microscope at 40X magnification so that uniformity and consistency may be ensured among all the cells. These whole slide image having thousands of cells is then passed through various preprocessing stages so to obtain individual segregated cells. Individual cells were cropped off from the slide image and then the color, brightness and contrast are enhanced using some image preprocessing techniques. These preprocessing steps makes the cytoplasm and nuclear region of the cell easily recognizable. In total 8091 individual cells were finally collected to create the final database. Each Indi dual cell were then manually tagged to actual label of diagnosis/class with the assistance of a trained pathologist as per the 2001 bathseda system of classification. In order to ensure a good quality each cell is inspected by two trained cyto-pathologists as per the class label attached to them. Any cell having any

kind of inconsistency is removed. Therefore the final database contains only those cells which are with accurate, precise and certain diagnosis. After preprocessing each cell is passed through a feature extraction module for the extraction of 39 morphological parameters. An open sourced module called CellProfiler written in python is used for the extraction of features. This module called CellProfiler is developed by BROAD institute and Massachusetts Institute of Technology to make the biologists enable enough to easily measure the phenotypes of medical images quantitatively. Fig 1. shows all the steps involved in creating this large database. After feature extraction from each cell a whole database containing 8091 tuples with 39 features in each tuple is created. This database is then used to train, test and validate the ten neural network algorithms used in the study. Table 1. summarizes the number of different types of Pap smear images of each class in the database. The first four classes i.e. Class 1, Class 2, Class 3 & Class 4 are classified as normal cells and the last three i.e. Class 5, Class 6 & Class 7 are classified as abnormal cells. Therefore, in two class classification problem all the first four classes are to be classified as benign category cells and the last three classes are to be classified as malignant category cell. Two class classification is always an easy classification task as compared to seven class classification. The screening of these Pap smear images can be seen as classification of a cervical cell image into normal and abnormal category i.e. two-class problem followed by an elaborated and detailed classification into seven respective classes.

Table 1. Summary of images of each type of cells

Class	Category	Type of cell	Number of Cell images	Sub total
Class 1	Normal	Superficial squamous	1,474	4,956
Class 2		Intermediate squamous	1,354	
Class 3		Parabasal squamous	1,056	
Class 4		Basal squamous	1,072	
Class 5	Abnormal	Low-grade squamous intraepithelial lesion	1,002	3,135
Class 6		High-grade squamous intraepithelial lesion	1,122	
Class 7		Squamous cell carcinoma	1,011	

2.2. Algorithms used for analysis

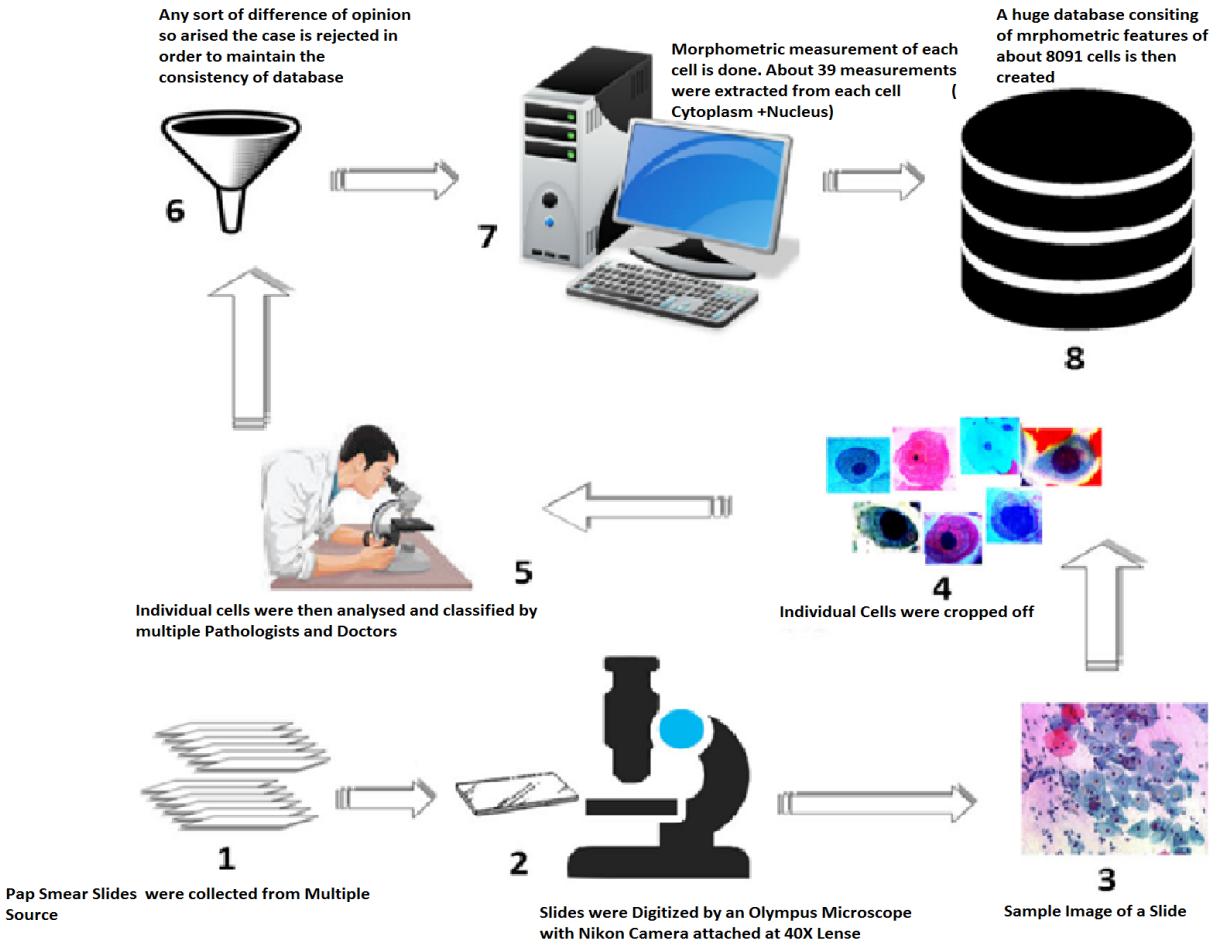

Fig. 1. All Steps involved in Creating a huge database of Cervical Cancer having 8091 Cells with 9 Morphological Features

In this work the authors have analyzed working and classification potential of ten well know neural network algorithms for classification of cervical cancer cases. Artificial Neural Networks also referred as neural networks is a mathematical formulation of human brain and have properties same as that of biological neural systems. Unlike the digital model of

computer, artificial neural networks works by manipulating connections between processing elements called neurons. Like the biological system, an artificial neural network learns from the external environment and shows a remarkable ability to learn, recall, generalize and adapt with respect to the external environment. A biological neural system contains billions of neurons connected with each other. Each neuron has synapses, dendrites, cell body and axon. Synapses are biochemical in nature and are responsible for converting pre-synaptic electrical signal to chemical signal and also produce post-synaptic electrical signal. Synapses can either be inhibitory or excitatory. Cell body of a neuron generates neuronal signal called spike. This signal is carried by the axon to the junction of synaptic terminal of other neurons. The frequency of firing of a neuron depends on the total synaptic activities and is controlled by the synaptic parameters. A neuron receives signals from other neurons in its neighborhood through dendrites. The neuron combines these excitations from all the dendrites and when the sum exceeds a certain threshold it fires by generating a spike. This spike or signal is passed to other neurons through axons. At the end of each branch of axon, synapses are present which carries the signal to the connected dendrites of other neurons. All the key features of neural network are borrowed from biological neural system. These key features include independent neurons, distributed memory, weights and updation of weights with respect to experience. An artificial neural network therefore consists of many independent processing elements called neurons. These neurons are connected with each other through weighted connections. These weights do actually carry the knowledge or the information of a neural network. These weights are initialized randomly or from some distribution during the first time. The objective of learning algorithms is to find a set o weights so that the error becomes minimum. Therefor the learning or training algorithm tries to adjust weights between the nodes in such a way so that the final error between what the network is computing and what is the actual output is gets minimum. After training/making the algorithm learn, the algorithm is tested on some unseen patterns to access its ability of prediction (Haykin, 1998). ANN can also deal with incomplete and noisy data (Mahamud et al.,1999)

A short description about each algorithm is given in below sections.

4.3.2.1 BFGS Quasi Newton Neural Network Algorithm

The basic aim of neural network learning is to make the set of weights get optimized. This algorithm uses Newton's method for learning/training. Newton's method is an alternate way of

optimization in neural networks to the basic conjugate gradient techniques. A Newton method is generally written as

$$X_{k+1} = X_k - A_k^{-1} g_k$$

where A_k^{-1} is the Hessian matrix (second derivatives) of the performance index. This hessian matrix is for the current set of weights and biases. One of the advantages of newton methods is that it converges faster than the normal conjugate gradient method. But Newton method is complex and computationally very costly and the Hessian Matrix computation is very time consuming. Some feed forward neural networks trained by Newton's method of optimization do not require the computation of second derivatives. These methods are called quasi-Newton (or secant) methods. Instead of using and updating a Hessian Matrix they updates an approximate version of Hessian Matrix at each iteration of the algorithm. This algorithm requires more time and storage for computation as compared to conjugate gradient methods but converges faster in fewer iterations. The approximate Hessain matrix used in the algorithm can be stored in a $N * N$ where N is the number of weights and biases in the network. For large networks with large number of weights it is better to use conjugate gradient methods as the size of hessian matrix will be very large for Newton Methods. The size of hessian matrix will make the computation of hessian very slow. For smaller networks the Quasi-Newton Method is good as the size of Hessian Matrix remains manageable. The BFGS algorithm can train any neural network if its weights, the oinput and the gtransfer function have a derivative function. The Backpropgation method is applied to calculate performance function derivative with respect to weights and bias. Each variable of the network is adjusted according to the following function. $X = X + a * dX$, dX is the search direction. The parameter a is used to minimize the performance function. A search line function is used to find the minimum point of performance function. At first the search direction is negative of the gradient of performance function. Later on in each iteration the direction of search is computed by the given formula $dX = \frac{-H}{gX}$ where gX is the gradient and H is the approximate Hessian Matrix.

.

2. 2.1. Fletcher-Powell Conjugate Gradient Algorithm

On the first iteration a conjugate gradient algorithm initiates its searching in the steepest

descent direction (negative of the gradient). $P_0 = -g_0$ Then to find the optimal distance to move along the search direction this equation is applied $X_{k+1} = X_K a_K P_K$. The new search direction is then calculated and the next search direction will be conjugate of previous search directions. In order to find the next new search direction this algorithm combines the steepest descent direction with the previous search direction. Given by$P_k = g_k + \beta_k P_{k-1}$. There are various versions for conjugate gradient algorithm and are distinguished by the manner in which the constant β_k is computed. The Fletcher-Reeves algorithm uses the given update rule$\beta_k = \frac{g_k^T g_k}{g_{k-1}^T g_{k-1}}$. This ratio is the ratio of norm squared of the gradient to the norm squared of the previous gradient. This algorithm is much faster than backpropagation with variable learning rate and is also sometimes faster than Resilient Backpropagation algorithm though the results vary from problem to problem. This algorithm is very useful for training large network as it requires a little more storage as compared to other simpler algorithms.

This algorithm can train any network as long as its weights, input and the transfer functions are differentiable functions. Backpropagation technique is applied to calculate the derivate of the performance function with respect to the weights and bias variable X .Each variable is adjusted according to the following equation $X = X + a * dX$ where dX is the search direction. A line search function is used to locate the minimum point of the performance function. In each step the search direction is calculated by using the gradient of new and previous search directions using the equation$dX = -gX + dX\ \ old * Z$ where gX is the gradient. The parameter Z can be computed in various ways and for the Fletcher-Reeves conjugate gradient algorithm. The computation of Z is given by $Z = \frac{normnew\_sqr}{norm\_sqr}$. Where $norm\_sqr$ is the norm square of the previous gradient and $normnew\_sqr$ is the norm square of the current gradient. The training of network in this algorithm stops if any of these conditions are met

- When the epochs reaches to maximum number.
- When the training time exceeds maximum amount of time.
- When the learning reaches to minimum error goal.
- The performance gradient falls below min_grad
- Validation performance has increased more than max_fail times since the last time it decreased (when using validation)

2.2.2. Polak-Ribiére Conjugate Gradient Algorithm

This algorithm is another version of conjugate gradient algorithm and was first proposed by Polak and Ribiére. The search direction at each iteration in case of Fletcher-Reeves algorithm is given by $P_k = g_k + \beta_k P_{k-1}$ the same search direction at each iteration in this algorithm is given as $\beta_k = \frac{\Delta g_{k-1} g_k}{g_{k-1}^T g_{k-1}}$. This equation of direction search computes the inner product of the previous change in the gradient with the current gradient divided by norm squared of the previous gradient. The time requirement is similar to Fletcher-Powell Conjugate gradient but the storage is little bit larger because this algorithm has four vector calculation as compared to Fletcher-Powell Conjugate gradient having only three vectors. The training phase of this algorithm is done through backpropagation method. The backpropagation method is used to calculate the derivative of the performance function with respect to the weight and the bias variable. Each weight and bias variable is adjusted as per the given equation $X = X + a * dX$ where dX is the search direction. Parameter a is selected in such a way to minimize the performance function along the search direction. The objective is to find the global minima for the performance function. A line search function is used to minimize the error function and locate the minimum point. The first search direction is always negative of the gradient of the performance function. In each iteration the next serach direction is computed from the new gradient and the previous gradient $dX = -gX + dX\ \ old * Z$ where gX is the gradient. The parameter Z can be computed in various ways and for the Polak-Ribiére Conjugate Gradient Algorithm the computation of Z is given by $Z = \frac{(((gX - gX\_old))' * gX)}{norm\_sqr}$ where $norm\_sqr$ is the norm square of the previous gradient and gX_old is the gradient of the previous iteration.

The training of network in this algorithm stops if any of these conditions are met

- When the epochs reaches to maximum number.
- When the training time exceeds maximum amount of time.
- When the learning reaches to minimum error goal.
- The performance gradient falls below min_grad
- Validation performance has increased more than max_fail times since the last time it decreased (when using validation).

2.2.3. Levenberg-Marquardt Algorithm

This algorithm was designed to achieve the second order training speed without having to

compute the Hessian Matrix like that of quasi-newton approaches (Marquardt D). This algorithms uses an approximate version of the hessian matrix rather a fully computed hessian matrix. The algorithm is faster as compared to simple backpropagation algorithms but requires large amount of memory for the hessian matrix. As usually the feed forward neural networks have the performance function as in form sum of squares then the hessian matrix can be approximated as $H = J^T J$ and the gradient of the function can be computed by $g = J^T e$. The J is the jacobian matrix and contains the first derivative of the network error with respect to the weights and biases. The e is the vector storing the error values. The jacobian matrix J is computed by using a standard backpropagation technique and is very less complex as compared to computation of Hessian matrix. This algorithm uses this approximation function to calculate an approximate version of hessian matrix by using the given equation. $X_{k+1} = X_K - [J^T J + \mu I]^{-1} J^T e$ when the scalar value μ is zero, then the equation is just like Newton's method and when μ is large the algorithm behaves like gradient descent with a small step size. The Newton's method is faster and accurate with minimum errors therefore the algorithm aims to shift towards the Newton's method as quickly as possible. Therefore the μ is decreased whenever step gets success and is increased when the performance function degrades. This way the algorithm tries to reduce the performance function at each iteration of the algorithm. This algorithm has the fastest training potential for feedforawrd neural networks of medium size with up to several hundred weights. This algorithm has also an efficient implementation in matlab. The Levenberg-Marquardt Algorithm is very efficient as compared to conjugate gradient algorithm when the network has only few hundred weights (Hagan, M.T., and M. Menhaj). Backpropagation algorithm is used to calculate the jacobian matrix jX of the performance function with respect to the weight and bias variable X. The algorithm adjusts each variable as $jj = jX * jX$, $je = jX * E$ and $dX = -(jj + I * \mu)/je$ Where E is all errors and I is the identity matrix. The value of μ is increased until by μ_inc until the change in above leads to reduction in performance function. The learning function stops when any of these conditions occurs when 1) maximum number of epochs reaches or 2)the training time exceeds or 3) performance function reaches the minimum possible point or 4) μ exceeds μ_max or 5) Validation performance has increased more than max_fail times since the last time it decreased (when using validation).

2.2.4. One Step Secant Algorithm

BFGS Quasi Newton Neural Network Algorithm uses more storage and computation time in

each iteration as compared to the conjugate gradient algorithms (Battiti, R.) In such algorithms there is always a calculation of secant approximation with smaller storage and computation. The One Step Secant Algorithm attempts to bridge the two types of algorithms one type is conjugate gradient algorithms and the other type is quasi-Newton (secant) algorithms. The greatest advantage of this algorithm is that it do not requires hessian matrix rather it assumes that the previous step hessian matrix as an identity matrix. Another advantage of this algorithm is that there is no need of computing the matrix inverse during the calculation of new search direction. The algorithm has an edge over BFGS algorithm as it requires less storage and computation during each epoch and requires more storage and storage per epoch as compared to conjugate-gradient algorithm. Therefore this algorithm can be seen as a compromise between the more time and storage required quasi-Newton algorithms and less storage and time required conjugate gradient algorithms. Backpropagation technique is applied to calculate the derivate of the performance function with respect to the weights and bias variable X .Each variable is adjusted according to the following equation $X = X + a * dX$ where dX is the search direction. A line search function is used to locate the minimum point of the performance function. In succeeding iterations the search direction is calculated by using the new gradient and the previous step and gradient using the equation $dX = -gX + Ac * X_{step} + Bc * dgX$ where gX is the gradient, X_{step} is how much weight is changed in the previous step and dgX is how much gradient is changed from the last iteration. This algorithm trains the feed forward neural network till any one of the conditions are met.

- When the epochs reaches to maximum number.
- When the training time exceeds maximum amount of time.
- When the learning reaches to minimum error goal.
- The performance gradient falls below min_grad
- Validation performance has increased more than max_fail times since the last time it decreased (when using validation).

2.2.5. Extreme Learning Machine

This algorithm is able to train feed forward neural networks with single hidden layer for classification and regression problems. The whole training process in this algorithm is finished in one iteration instead of many iterative steps of training like that of other well-known and previously discussed algorithm. As the algorithms only need one iteration for whole training purpose this algorithm is very fast for training. In comparison to other gradient descent based

neural network this algorithm is almost ten times faster **(Deng et al. 2010a, b).** This algorithm can randomly set the input weights and biases of the hidden layer. The weights of output layer nodes are obtained through least square method. The weights between input nodes and hidden nodes are randomly assigned and are kept constant during the training process. In the learning phase only the weights between hidden and output nodes are trained at a very fast rate. Feed forward network trained by ELM algorithm produces acceptable performance and the cost of training and maintaining a network is very low as compared to standard backpropagation neural networks. This algorithm is very easy and effective for single hidden layer feed forward neural networks as the traditional algorithms requires to set lot of parameters related to network training and these parameters if not adjusted optimally leads to local optima. In ELM we need to set only the number of hidden nodes in the hidden layer and the training algorithm do not need to train/learn the input weights as well as hidden layer biases. Therefore this algorithm is able to generate unique optimal solution with the advantage of fast training and greater generalization power. A dataset with N distinct samples and each sample is represented as (x_i, t_i) where $x_i = [x_{i1}, x_{i2}, x_{i3}, \dots\dots\dots x_{in}]^T \in R^n$ *and* $t_i = [t_{i1}, t_{i2}, t_{i3}, \dots\dots\dots t_{im}]^T \in R^m$. A single layer feed forward neural network with K number of hidden nodes models mathematically as

$\sum_{i=1}^{K} \beta_i f_i(x_j) = \sum_{i=1}^{k} \beta_i f(a_i . x_j + b_i) = t_j, j = 1 \dots N$ Where

$a_i = [a_{i1}, a_{i2}, a_{i3}, \dots\dots\dots a_{in}]^T$ represents the weight vector. This weight vector connects the i^{th} hidden node and the input node.

. $\beta_i = [\beta_{i1}, \beta_{i2}, \beta_{i3}, \dots\dots\dots \beta_{im}]^T$ vector stores the weight values between the connecting the i^{th} hidden node and the output node . the activation function can be sine, sigmoid and radial basis function. The above equations can be written in Matrix form as

$H\beta = T$ where the $H = (a_1, \dots . a_K, b_1, \dots . b_K, x_1, \dots . x_N) = [f(a_1 . x_1 + b_1)\ f(a_k . x_1 +$ *bk .... fai.xN+b1 fak.xN+bK* and β=*β1T .... βkT* , *T=t1T .... tkT*. The matrix H represents the hidden layer output matrix of the ELM Network. for a good classification performance the $K \ll N$ is strongly recommended.

3. Literature review with respect of application of neural networks for detection of Cancer

(Van der Gaag et al 1992) developed a diagnostic tool for selecting a particular therapy for esophageal cancer as per the stage patient is suffering. This system assisted the oncologists in

finding the correct stage of cancer and planning the proper treatment plan on time. The kernel of system was probabilistic network. This probabilistic network described characteristics of the esophagus cancer and the patho-physiological processes of invasion and metastasis. The authors have designed a new method of probability elicitation in this system by combining the idea of transcribing probabilities as fragment of text and also using a scale with both numerical and verbal anchors for the assessment. This idea of combination allowed the authors to elicit any probabilities in a very reasonable time. For the evaluation purpose they have applied the system on real patient data and the system produced correct stage of cancer for 85% patients. **(Liu et al 1996)** designed a tree based machine learning algorithm named as PREDICTOR. This algorithm produced some rules to classify a patient as either suffering from esophageal cancer or normal. To train and validate the algorithm the authors have created a database 2692 dyspeptic patients having 73 attributes (epidemiological and clinical). All the patients were above 40 years of age. The authors have also constructed an expert system based on rules extracted from the database by the PREDICTOR. This expert system is used to detect cancer in dyspeptic patients. The results produced were very motivating. The cross validation results showed and classified 61.3% patients as high risk patients, with a sensitivity of 94.9% and specificity of 39.8%. **(Hammond et al 1998)** applied machine learning techniques for prognosis of oral cancer. The authors have demonstrated that these automated tools are performing at par with that of medical specialists in screening of oral cancer. The incidence of oral cancer is very limited but if kept untreated becomes very serious. Because of low rate incidence a widespread screening is not feasible therefore such automated computer aided tools can assist the health care professional. A patient screened by automated system can be recalled for further reexamination of oral mucosa and life style counselling. **(Sarkar et al)** studied the breast cancer diagnosis as a pattern recognition problem in computer science. Applied the publicly available Wisconsin-Medison Breast cancer dataset for training and testing of machine learning algorithms. K- Nearest neighbor algorithm is applied for classification, this algorithm produced significantly better results as compared to other machine learning techniques used for the same problem. On comparing it is found that the K-NN produced significantly results and is 1.17% better than the best known solutions. **(Delen et al 2005)** designed a system to predict the survivability of patient suffering from breast cancer. Two popular data mining algorithms artificial neural network and decision tree were used along with a statistical a technique called logistic regression are used to create the automated system. A secondary dataset consisting of

2, 00, 00 cases were used to train and test the system. Tenfold cross validation is applied to get an un-biased performance estimation. In tenfold cross validation the dataset is divided into 10 equal portions. These ten portions are then used to train and test the learning techniques ten times. Each time out of ten portions nine portions are used to train the algorithm and one portion is used to validate. The results indicated that the decision tree (C5) produced the best accuracy followed by the neural networks then logistic regression. This study produced an insight of prediction ability of different algorithms.

4. Methodology.

A detailed description of all the steps followed in experiment 1 is given is shown in fig.2.
Out of the ten neural network algorithms available we starts from the first training algorithm

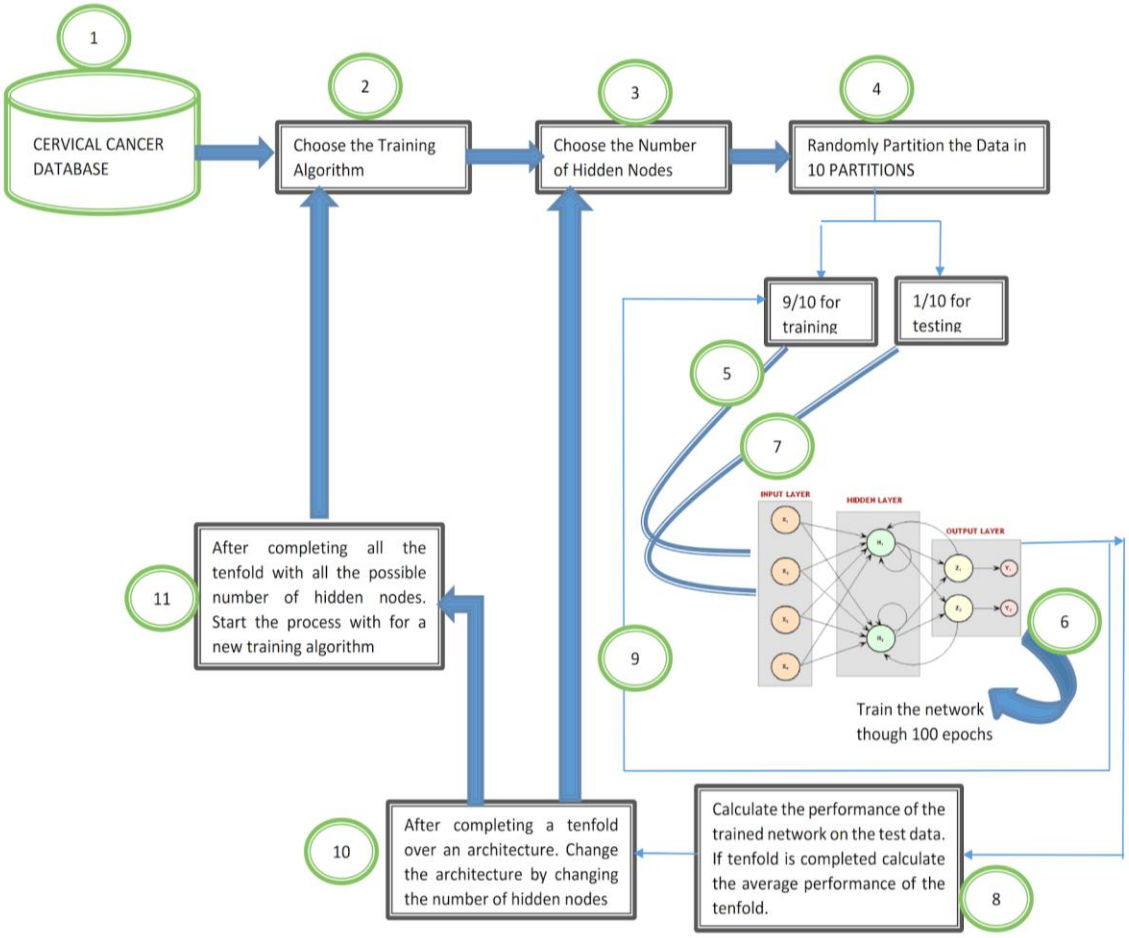

Fig.2. complete methodology

the number of hidden nodes in network is selected. The number of hidden nodes ranges from 2 to 20 for each training algorithm. In other words for each training algorithm we are creating 19 different networks with different number of hidden nodes staring from 2. After deciding the

proper architecture of the neural network the dataset is randomly partitioned into ten parts for tenfold cross validation. Out of the nine parts nine parts are used for training and 1 part out of ten part is sued for testing. The neural net is trained through 100 epochs in order to make the training algorithm sufficient time to train the weights. For every fold of tenfold the classification result of the network is evaluated by the then test set and the result is stored. After the tenth fold the final average value of the training algorithm with the given number of hidden nodes is calculated. The tenfold cross validation for each algorithm fitted with a given number of hidden nodes makes sure that the results so obtained are not biased and are very reliable. The process is repeated again for the same training algorithm with a different value for the hidden nodes. After evaluating all the hidden nodes for a training algorithm, a new training algorithm is selected and the process is started again. In this way a rigorous experiment is conducted to evaluate the classification potential of these ten neural network training algorithms on different performance parameters. The results so obtained are saved and analyzed for further enhancement. The per class classification accuracy for the seven class classification is evaluated and analyzed for further study. The class wise classification for seven class is done for all the ten algorithms on all the hidden nodes. In this way not only the classification potential of whole network is evaluated rather per class accuracy is also evaluated. The results of the best networks are also depicted through confusion matrix. The confusion matrix easily helps to calculate the false positive and false negative rates. A module is coded in Matlab for implementation of the whole experiment 1. The module has a separate function to divide the database into ten folds with equal chance of all classes in the dataset in order to avoid any kind of bias. The ten well known neural network algorithms available in the Matlab Neural network Toolbox are used and called though appropriate function calls providing the number of hidden nodes. The module tests the trained network by the toolbox on the test data of the folds and saves the result in an excel file. The Matlab toolbox do not have inbuilt extreme learning machine algorithm, therefore the authors have written a code in Matlab for the implementation of the same. A code is added to the module to perform the evaluation part. The evaluation code is able to compute and draw the confusion matrix. This evaluation code calculates and provides the class wise classification accuracy, class wise precision, recall and F-Value.

5. Results and Analysis

The experimental study 1 rigorously evaluates the classification potential of ten well known

monolithic neural networks. Each of these training algorithms are tested on different number of hidden nodes. In the analysis part the best performance of all the monolithic neural network is reported as. The classification accuracy vs the number of hidden nodes for all the algorithms for tenfold cross validation is shown in Fig.3. Among all the training algorithm the best three performers were Levenberg-Marquardt Algorithm followed by Variable Learning Rate Gradient Descent Algorithm and One Step Secant Algorithm. The Levenberg-Marquardt Algorithm produced a classification accuracy of 78.6%, Variable Learning Rate Gradient Descent Algorithm produced an accuracy of 77.3% and One Step Secant Algorithm displayed a classification potential of 76.2%. Among all the algorithms the Extreme learning Algorithm performs the least.

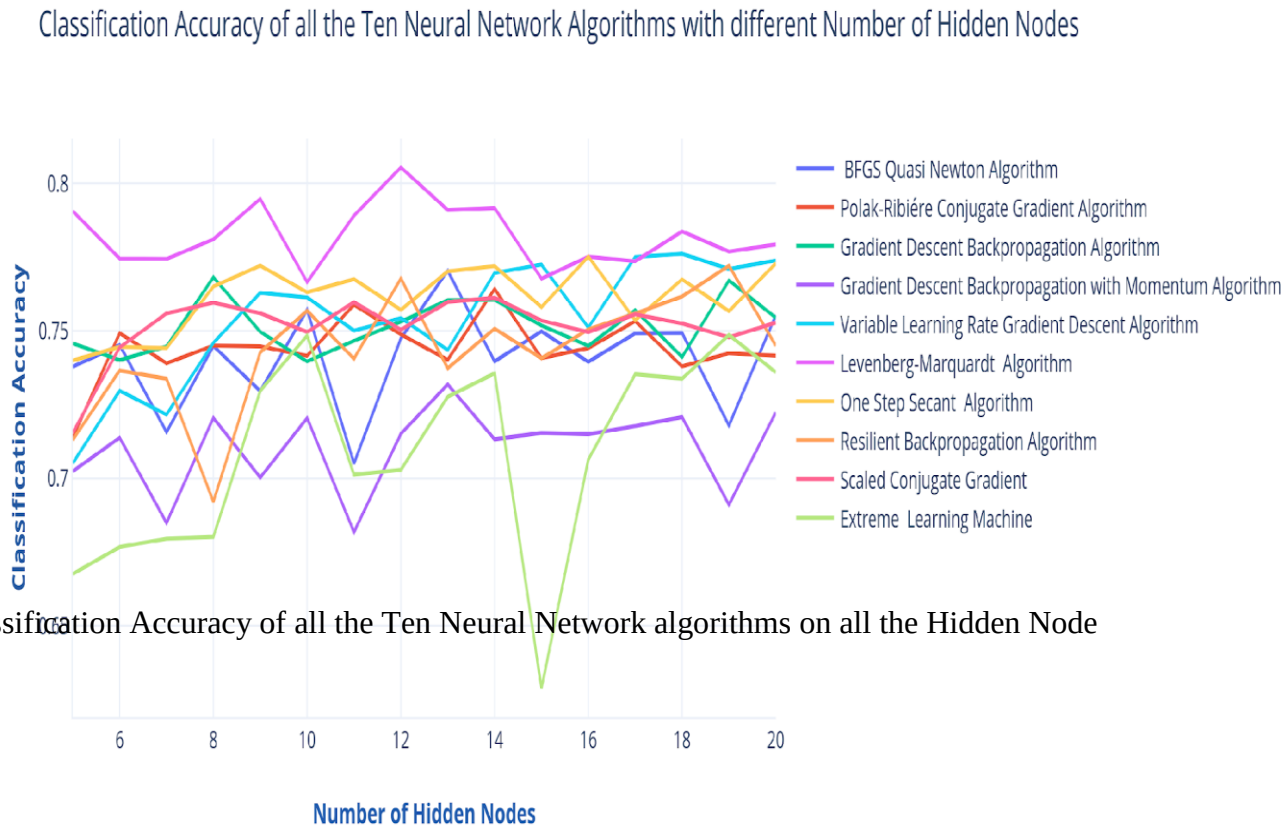

Fig.3. Classification Accuracy of all the Ten Neural Network algorithms on all the Hidden Node

International Journal of Computer Science & Communication (ISSN: 0973-7391)
Volume 17 • Issue 2 pp. 557-575 March 2026-Sept 2026 www.csjournals.com

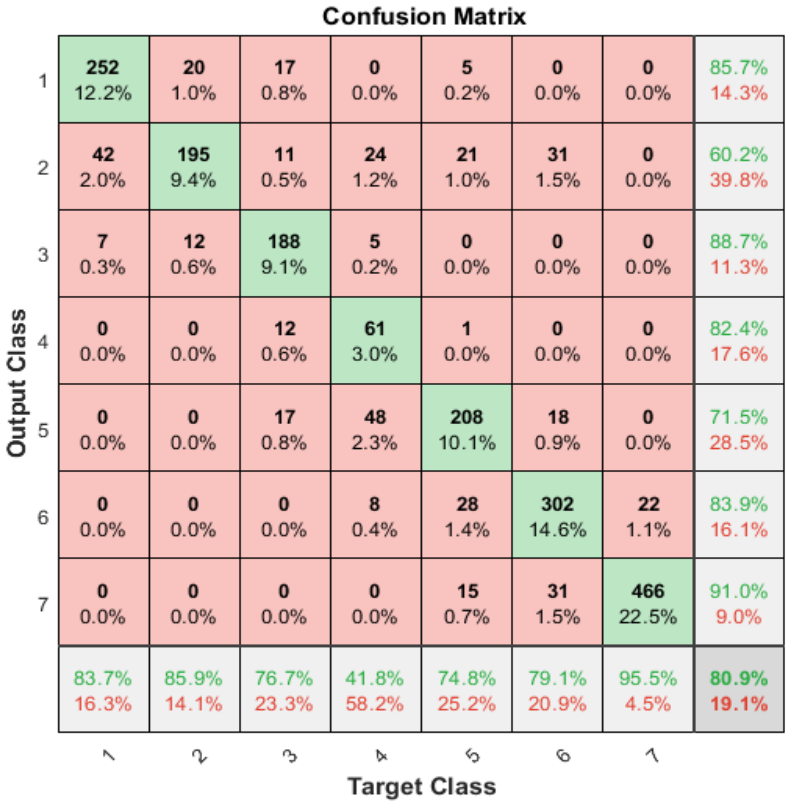

Fig.4. Confusion Matrix of Levenberg-Marquardt Algorithm with 12 hidden nodes

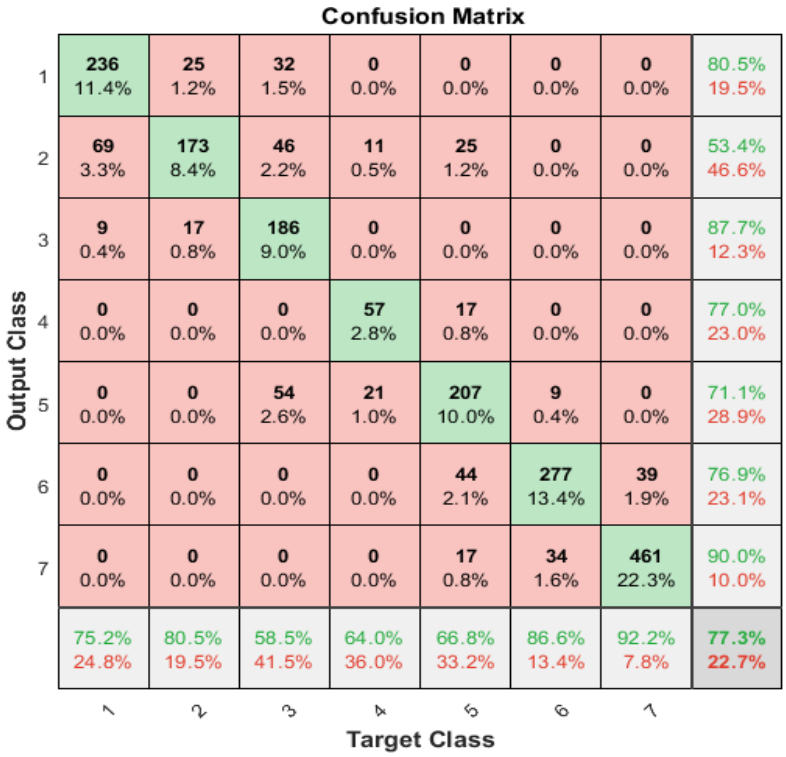

Fig.5. Confusion Matrix of Variable Learning Rate Gradient Descent Algorithm with 18 hidden nodes

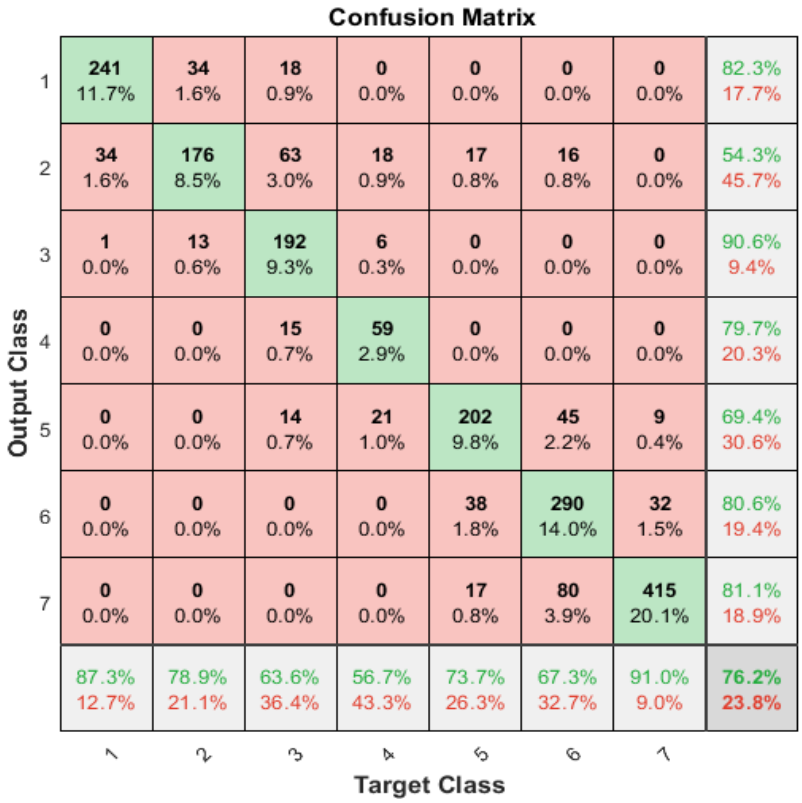

Fig.6. Confusion Matrix of One Step secant Algorithm with 16 hidden nodes

The confusion matrices of these three performing algorithms are shown in fig.4 to fig 6. The confusion matrices clearly depicts that the Class 2 produced the most misclassified rate as compared to all other classes followed by Class 5 in all the three algorithms. Class 2 belongs to the major group of Normal cases and Class 5 is from the abnormal group. The class 2 instances are misclassified into Class 3, Class 1, Class 4, Class 5 and even as Class 6 and among them Class 5 and Class 6 do belong to abnormal classes. Which means that a normal category Cell is Classified as abnormal category Cell thus increasing the rate of false positive. In both one Step Secant and Variable learning rate Gradient algorithm almost 50% of the Class 2 instances are misclassified. On the other hand all the three algorithms have misclassified the class 5 into Class 3, Class 4, Class 6 and class 7. Among these Classes Class 6 and Class 7 are from abnormal category and Class 3 and Class 4 are from Normal Category cells. Classifying a Class 5 Cell as either Class 3 or Class 4 means that an abnormal cell is classified as normal cell which means the positive case of cancer is diagnosed with negative. Therefore the misclassification of class 5 increases the false negative rate which is medically very costly. Among the three best performing algorithms the Levenberg Marquardt and Variable learning

rate gradient descent algorithm produced the greatest classification accuracy for Class 7 with classification accuracy of 91% and 90% respectively, thus making it clear that both these monolithic neural networks can classify the Class 7 better all other classes. Class 7 is the most deformed cell category and is at the most advanced stage of Cancer, therefore correct detection and classification of this category is very important for any diagnostic tool. Whereas the Variable learning rate gradient descent algorithm produced the best classification accuracy for the seven class. The one step secant produces the best classification result for class 3. A good diagnostic tool must have a very low false positive and false negative rate, therefore this study concludes that among the ten monolithic neural networks not a single algorithm even the best is not able to provide us a good classification result. Therefore, some more methods and techniques must be adopted in order to make these monolithic neural networks learn the problem space of cervical cancer classification in a more meaningful and better way.

References:

1. Marquardt, D., "An Algorithm for Least-Squares Estimation of Nonlinear Parameters," SIAM Journal on Applied Mathematics, Vol. 11, No. 2, June 1963, pp. 431–441.
2. Hagan, M.T., and M. Menhaj, "Training feed-forward networks with the Marquardt algorithm," IEEE Transactions on Neural Networks, Vol. 5, No. 6, 1999, pp. 989–993, 1994.
3. Battiti, R., "First and second order methods for learning: Between steepest descent and Newton's method," Neural Computation, Vol. 4, No. 2, 1992, pp. 141–166.
4. WY Deng, QH Zheng, L Chen, XB Xu, "Research on extreme learning of neural networks", Chinese Journal of Computers, 33(2):279–287, 2010
5. Van der Gaag, Linda C., et al. "Probabilities for a probabilistic network: a case study in esophageal cancer." Artificial Intelligence in medicine 25.2 (2002): 123-148
6. Liu, W. Z., White, A. P., Hallissey, M. T., & Fielding, J. W. L. (1996). Machine learning techniques in early screening for gastric and oesophageal cancer. Artificial intelligence in medicine, 8(4), 327-341
7. Hammond, Peter, and Paul M. Speight. "Screening for Risk of Oral Cancer and Pre-cancer." Intelligent Data Analysis In Medicine and Pharmacology (1998)
8. Sarkar, Manish, and Tze-Yun Leong. "Application of K-nearest neighbors algorithm on breast cancer diagnosis problem." Proceedings of the AMIA Symposium. American Medical Informatics Association, 2000.
9. Delen, Dursun, Glenn Walker, and Amit Kadam. "Predicting breast cancer survivability: a comparison of three data mining methods." Artificial intelligence in medicine 34.2 (2005): 113-127.