**Neural Network and Fuzzy Systems**

**Breast cancer detection using neural network**

**Zulqarnain Haider**

**14031269**

**Reviewed By**

**Abdul Qadeer Ahmad**

Contents

[Abstract 4](#_Toc500107288)

[Introduction 4](#_Toc500107289)

[Background 4](#_Toc500107290)

[Related Work 4](#_Toc500107291)

[Data Set 5](#_Toc500107292)

[Data Set Description 5](#_Toc500107293)

[Preprocessing 5](#_Toc500107294)

[Network Architecture 6](#_Toc500107295)

[Creating and training network 6](#_Toc500107296)

[Post Processing 6](#_Toc500107297)

[Overall functioning 6](#_Toc500107298)

[Experimental Results and Analysis 6](#_Toc500107299)

[List of Hypothesis 7](#_Toc500107300)

[Effect of transfer function 7](#_Toc500107301)

[Hypothesis 7](#_Toc500107302)

[Setup for Experiment 7](#_Toc500107303)

[Results of Experiments 7](#_Toc500107304)

[Effect of training Function 7](#_Toc500107305)

[Hypothesis 7](#_Toc500107306)

[Setup for Experiment 7](#_Toc500107307)

[Results of Experiments 7](#_Toc500107308)

[Effect of Hidden Layers 8](#_Toc500107309)

[Hypothesis 8](#_Toc500107310)

[Setup for Experiment 8](#_Toc500107311)

[Results of Experiments 8](#_Toc500107312)

[Effect 0f Neurons 8](#_Toc500107313)

[Hypothesis 8](#_Toc500107314)

[Setup for Experiment 8](#_Toc500107315)

[Results of Experiments 8](#_Toc500107316)

[Effect of data distribution 9](#_Toc500107317)

[Hypothesis 9](#_Toc500107318)

[Setup for Experiment 9](#_Toc500107319)

[Results of Experiments 9](#_Toc500107320)

[Best Result 9](#_Toc500107321)

[Conclusion 10](#_Toc500107322)

[References 10](#_Toc500107323)

# Abstract

Identification of breast cancer is an open study area which can be solved by autonomous systems. The ability of a software to self-direct itself was the idea that led the world to think of ‘autonomous systems’. The sole purpose of developing such systems was to make the software self-operative and self-dependent so as to minimize or remove human interaction. This report is about the breast cancer classification.

# Introduction

Breast cancer is widely spreading disease which needs to be tackled. Its second highest death reason among women. It is the most common cancer in women worldwide generating nearly 1.7 million new cases in 2012. This report gives a vision of how we can use a neural network for classifying cases of breast cancer using a backpropagation neural network. It covers all the stages from setting and designing a network to obtained and analyzed all the results from the network.

# Background

According to Dr. Robert Hecht-Nielsen who is the inventor of the first neurocomputer defines the neural network as ‘a computing system made up of a number of simple, highly interconnected processing elements, which process information by their dynamic state response to external inputs.’ (Artificial Intelligence Neural Networks, n.d.)

The neural network architecture is inspired by neurons in human brain. A neural network is consist of many interconnected neurons. There are two main classes of neural networks. The first one is Feed-forward networks and another one is Feedback network. First one is used as static and other is as dynamic.

# Related Work

Classification of breast cancer is a noticeable problem in the field of computing.

El-Sebakhy et al. (2006) work on this problem in which Functional networks were proposed as a classification scheme for the breast cancer classification and found the accuracy of 96.8%. (Evaluation of Breast Cancer Tumor Classification with Unconstrained, 2006).

Senapati et. al(2013) found the classification accuracy of 98.14% to 99.6%, 97.4% accuracy in Malignant and Benign tumors respectively. (Senapati, 2012)

# Data Set

The dataset was taken from UCI Machine Learning dataset repository. Data contain 11 column and 699 rows and there were 16 instances where the values are missing and denoted by ‘?'.

# Data Set Description

As describe above dataset consists of 699 rows that are the number of instances and 11 column in which the first column is for the ID of the instance and next 9 are the input attributes for the breast cancer dataset. The end column gives the result that from which class this instance linked.

Classes are of two types:

* Benign (2 for Benign)
* Malignant (4 for Malignant)

In given dataset, 65.5% instance (458) are distributed in benign class and 34.5% instance (241) are for malignant class.

Following are the nine attributes of input data and their values are in between from 1 to 10.

* Clump Thickness
* Uniformity of Cell Size
* Uniformity of Cell Shape
* Marginal Adhesion
* Single Epithelial Cell Size
* Bare Nuclei
* Bland Chromatin
* Normal Nucleoli
* Mitoses

# Preprocessing

We have 699 instances and 11 columns. The first column is ID number that is useless for experiments so we need to remove that or neglect it. The last column is the class of instance that is denoted by 2 for benign and 4 for malignant. Here, first of all, replace the missing instance's value. According to suggested method of E. Pesonen et al. we replaced it with the mean value of the column. For this purpose following lines of code will work

trainingInput(trainingInput==0) = round(mean(trainingInput(:,6)));

Frist load the dataset.

data = load('dataset.data');

After that, we divide the data set into training and testing ration. So here I can divide it into two sets of 70 and 30 ration for training and testing respectively. It may vary in some of the hypothesis but there it will be clearly mention. Following lines of code divides the data into training input of first 70% and output class too.

trainingInput = data(1:489,2:end-1);

training utput = data(1:489,end);

This is the remaining data which will be tested on 30% data.

testingInput = data(490:699,2:end-1);

testingOutput = data(490:699,end);

# Network Architecture

Most importantly, for the characterization of Breast Cancer, Feed Forward Neural Network, a notable engineering of Neural Network is utilized with the assistance of a Back-Propagation calculation. Also, there is a specific rule to identify various neurons in the hidden layers. Subsequently, to get the most precise outcome, distinctive quantities of neurons in hidden layer were utilized for the examination.

# Creating and training network

After all preprocessing when the informational index was prepared a neural system was manufacture utilizing newff work in Matlab then its distinctive characteristic was set like learning rate, objective, epochs, initiation work and so forth. When arrange design was fabricated it was sent to prepare work for preparing information so as to prepare the system. In the wake of preparing, it was then sent to 'net capacity of Matlab with testing data to test the network.

# Post Processing

After all the training and testing function we have to need to the efficiency or accuracy of the network. For that reason, we compare both testing output (Class column) and the expected output and then divide by the total number of instances and then multiply it by 100 for a percentage.

accuracy = testingOutput - expectedOutput';

match\_index = sum(accuracy(:)==0);

percentage = (match\_index/total instances \* 100);

# Overall functioning

# Experimental Results and Analysis

All results are based on the hypothesis. Moreover, experiments performed with 3000 epochs, 0.01 goal, max\_fail of 100 points and default performance function of mean square error (me). As far as the learning rate (or) is concerned, the larger the learning rate, the bigger the step. Hence, as suggested by (Hagan, M, et al.1996), an average value of 0.02 was selected for all experiments.

# List of Hypothesis

## Effect of transfer function

tansig and purelin are the two transfer function. Lansing has a sine wave and pure has a straight line of its output values.

### Hypothesis

tansig should perform best because of the partial sine wave that can help in fast learning.

### Setup for Experiment

Here I use the one hidden layer with 10 neurons and the dataset is divided into two sets of 70/30 ration. I use training function trainbr for checking this experiment.

### Results of Experiments

|  |  |  |  |
| --- | --- | --- | --- |
| Transfer function | Training function | Neurons (in HL) | Accuracy |
| Tansig, tansig | trainbr | 10 | 97.61 |
| Tansig, purelin | Trainbr | 10 | 96.66 |
| Purelin, tansig | Trainbr | 10 | 95.71 |
| Purelin, purelin | trainbr | 10 | 90.00 |

#### Conclusion of Experiment

From the above experiment, it shows that tansig give the more accuracy then others also it is not time-consuming. Now here I am sure that tansig is the function which gives the more accuracy now I deduce one more hypothesis from this experiment.

## Effect of training Function

As from above hypothesis, it showed that tansig function gives more accuracy than other, now here we check the different training algorithms. As trainbr gives the more accuracy because trainbr is a network training function that updates the weight and bias values according to Levenberg-Marquardt optimization. It minimizes a combination of squared errors and weights and then determines the correct combination so as to produce a network that generalizes well.

### Hypothesis

Trainbr training function gives more accuracy because it minimizes the combination of squared errors and weights.

### Setup for Experiment

Here I use the one hidden layer with 10 neurons and the dataset is divided into two sets of 70/30 ration. I use tansig transfer functions for the experiment.

### Results of Experiments

|  |  |  |  |
| --- | --- | --- | --- |
| Transfer function | Training function | Neurons (in HL) | Accuracy |
| Tansig, tansig | trainbr | 10 | 98.09 |
| Tansig, tansig | trainlr | 10 | 92.10 |
| Tansig, tansig | trainbfg | 10 | 97.14 |
| Tansig, tansig | trainrp | 10 | 95.71 |
| Tansig, tansig | trainscg | 10 | 96.66 |
| Tansig, tansig | traincgb | 10 | 96.19 |
| Tansig, tansig | traincgf | 10 | 96.66 |
| Tansig, tansig | traincgp | 10 | 96.19 |
| Tansig, tansig | trainoss | 10 | 94.28 |
| Tansig, tansig | traingdx | 10 | 94.76 |

#### Conclusion of Experiment

Trainbr function gives the most accuracy in same flow with the tansig transferring function.

## Effect of Hidden Layers

### Hypothesis

Increasing number of hidden layers should increase learning time and accuracy will also increase.

### Setup for Experiment

Everything keeps constant for the experiment only the number of hidden layer increases. Nodes in each hidden layer are 10 and transferring function tansig and training function tanbr is use. All the experiment will perform on 70/30 ratio of the dataset.

### Results of Experiments

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Hidden Layers | Transferring function | Training function | Neurons (in HL) | Accuracy |
| 1 | Tansig, tansig | trainbr | 10 | 98.09 |
| 2 | Tansig, tansig | Trainbr | 10 | 96.66 |
| 3 | Tansig, tansig | Trainbr | 10 | 95.71 |
| 4 | Tansig, tansig | trainbr | 10 | 97.14 |
| 5 | Tansig, tansig | Trainbr | 10 | 0 |

#### Conclusion of Experiment

The experiment went wrong surprisingly by increasing hidden layers the accuracy was decreasing.

## Effect 0f Neurons

From the previous hypothesis, we come to know that one hidden layer gives us more accuracy.

### Hypothesis

Increasing the neurons in hidden layers also increases the accuracy.

### Setup for Experiment

Only one hidden layer is used for changing a number of neurons, transferring functions are tansig and training function is trainbr. All the experiment will perform on 70/30 ratio of the dataset.

### Results of Experiments

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Hidden Layers | Transferring function | Training function | Neurons (in HL) | Accuracy |
| 1 | Tansig, tansig | trainbr | 8 | 97.14 |
| 1 | Tansig, tansig | trainbr | 10 | 97.61 |
| 1 | Tansig, tansig | Trainbr | 12 | 98.57 |
| 1 | Tansig, tansig | Trainbr | 14 | 97.61 |
| 1 | Tansig, tansig | trainbr | 16 | 98.57 |
| 1 | Tansig, tansig | Trainbr | 18 | 96.66 |

#### Conclusion of Experiment

By increasing hidden layers it will effect on accuracy it will increase by increasing number of neurons but to some extent so the 12 neurons in one hidden layer are best option.

## Effect of data distribution

By all the above hypothesis we conclude the following results

* Tansig transferring function gives more accuracy.
* Trainbr training function is best for this type of dataset.
* Only one hidden layer with 12 neurons gives the maximum accuracy.

All these conclusions are based on 70/30 ratio of dataset distribution.

### Hypothesis

Increasing the training data from low to high more accurate value.

### Setup for Experiment

Transfering functions are tansig while training function is trainbr with one hidden layer and 12 neurons. Data distribution is change throughout the experiment.

### Results of Experiments

|  |  |  |
| --- | --- | --- |
| Training data | Testing data | Accuracy |
| 10% | 90% | 93.00 |
| 20% | 80% | 95.34 |
| 30% | 70% | 95.24 |
| 40% | 60% | 95.46 |
| 50% | 50% | 95.70 |
| 60% | 40% | 96.41 |
| 70% | 30% | 99.52 |
| 80% | 20% | 96.65 |
| 90% | 10% | 98.55 |

#### Conclusion of Experiment

On this setup, we'll conclude that the if we train the data on 70% and test on remaining 30% dataset it gives the more accuracy.

# Best Result

Highest accuracy of 99.5% accuracy was obtained using only one hidden layer with 12 neurons, tansig, tansig as a transfer function and trainbr as the training function. Overall, 99.5% of the predictions are correct and 0.5% are wrong classifications.

# Conclusion

This work was focused on the usage of different approaches to neural networks to achieve the best result. It experimented that the accuracy is dependent on the data distribution for training as well as testing. By using the fair distribution of 70/30, 99.5% accuracy was achieved. It was also analyzed that the different learning algorithms have some impact on results, not only in terms of training speed but also on the accuracy, as analyzed that the quickest learning rate, as well as the highest accuracy of 98.09%, was achieved with the trainbr training function. Proposed neural network with one hidden layer of 12 neurons using trainbr training function can be used to achieve a better classification accuracy. In any case, the general performance would, in any case, be influenced by some different variables.

# References

Artificial Intelligence Neural Networks. (2017, 11 2). Retrieved from Tutorials Point: https://www.tutorialspoint.com/artificial\_intelligence/artificial\_intelligence\_neural\_networks.htm

Evaluation of Breast Cancer Tumor Classification with Unconstrained. (2017, 11 3). Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.567.3856&rep=rep1&type=pdf

E. Pesonen, M. Eskelinen, and M. Juhola. Treatment of missing data values in a neural network based decision support system for acute abdominal pain. Artificial Intelligence In Medicine, 1998

E. A. El-Sebakhy, K. A. Faisal, T. Helmy, F. Azzedin, A. Al-Suhaim, et al. Evaluation of breast cancer tumor classification with unconstrained functional networks classifier, 2006.

M. R. Senapati and P. K. Dash. Local linear wavelet neural network based breast tumor classification using firefly algorithm. Neural Computing and Applications, 22(7-8):1591-1598.