Expert Systems with Applications 42 (2015) 4611–4620

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Breast cancer diagnosis using Genetically Optimized Neural Network model

Arpit Bhardwaj *, Aruna Tiwari

Computer Science and Engineering Department, Indian Institute of Technology Indore, India

ARTICLE INFO

Article history:
Available online 4 February 2015

Keywords:
Genetically Optimized Neural Network
Artificial Neural Network
Genetic Programming
Modified Crossover Operator

ABSTRACT

One in every eight women is susceptible to breast cancer, at some point of time in her life. Early detection and effective treatment is the only rescue to reduce breast cancer mortality. Accurate classification of a breast cancer tumor is an important task in medical diagnosis. Machine learning techniques are gaining importance in medical diagnosis because of their classification capability. In this paper, we propose a new, Genetically Optimized Neural Network (GONN) algorithm, for solving classification problems. We evolve a neural network genetically to optimize its architecture (structure and weight) for classification. We introduce new crossover and mutation operators which differ from standard crossover and mutation operators to reduce the destructive nature of these operators. We use the GONN algorithm to classify breast cancer tumors as benign or malignant. To demonstrate our results, we had taken the WBCD database from UCI Machine Learning repository and compared the classification accuracy, sensitivity, specificity, confusion matrix, ROC curves and AUC under ROC curves of GONN with classical model and classical back propagation model. Our algorithm gives classification accuracy of 98.24%, 99.63% and 100% for 50–50, 60–40, 70–30 training–testing partition respectively and 100% for 10 fold cross validation. The results show that our approach works well with the breast cancer database and can be a good alternative to the well-known machine learning methods.

1. Introduction

The cell is the basic biological unit of all living organisms, including humans. A normal cell, in its life cycle, grows in size, during which it collects nutrients, and then divides itself to form two new daughter cells. Cells become cancerous when they lose their ability to stop dividing, to stay where they belong, and to die at the proper time. Such extra cells together form a mass called tumor. The tumor can be identified either as benign or malignant. Breast cancer is a malignant tumor originating from the breast tissue (Muto, Bussey, & Morson, 1975).

Every woman is at risk for breast cancer. If she lives to be 85, there is a one in eight chance (12%) that she will develop breast cancer sometime during her life. As a woman ages, her risk of developing breast cancer rises dramatically regardless of her family history. Treatment and causes for breast cancer are still under research and since there is no widely available preventive measure (Christoyianni, Dermatas, & Kokkinakis, 2000; Rodrigues, Giraldi, Chang, & Suri, 2006) yet, early detection and effective treatment is the only rescue to reduce breast cancer mortality. Fortunately, if breast cancer is detected accurately in an early stage, localized tumors can be treated successfully before the cancer spreads. Thus, accurate diagnosis of breast cancer is an essential and urgent problem in medical science community.

One of the major task for accurate diagnosis is the extraction of useful knowledge from past diagnosis data. Machine learning techniques enable computers to learn from experience, past patterns and examples (Carbonell, 1983; Witten & Frank, 2005). Thus, use of machine learning tools in medical diagnosis is increasing gradually. Data mining and soft computing techniques have been applied to extract rules and patterns from various datasets (Maimon & Rokach, 2005;Mitra & Hayashi, 2000;Mitra & Acharya, 2005). Some of these techniques (Bellazzi & Zupan, 2008;Marcano-Cedeño, Quintanilla-Domínguez, & Andina, 2011;Malmir, Farokhi, & Sabbaghi-Nadooshan, 2013) have shown very good results in classification problems, which can help medical experts in recognizing diseases. In case of breast cancer diagnosis, a tumor needs to be identified as benign or malignant based on the sample properties. In terms of machine learning, this problem can be approached as a 2-class classification problem based on a set of sample attributes.

A wide range of methods have been proposed to forecast medical diagnosis of breast cancer with WBCD in literature. Quinlan

* Corresponding author.
E-mail addresses: phd12110102@iiti.ac.in (A. Bhardwaj), artiwari@iiti.ac.in (A. Tiwari).

http://dx.doi.org/10.1016/j.eswa.2015.01.065
0957-4174/© 2015 Elsevier Ltd. All rights reserved.

Table 1
Description of Wisconsin Breast Cancer Database.

Attribute number	Attribute	Values	Mean	Standard deviation
1	Clump thickness	1–10	4.44	2.83
2	Uniformity of cell size	1–10	3.15	3.07
3	Uniformity of cell shape	1–10	3.22	2.99
4	Marginal adhesion	1–10	2.83	2.86
5	Single epithelial cell size	1–10	2.23	2.22
6	Bare nuclei	1–10	3.54	3.64
7	Bland chromatin	1–10	3.45	2.45
8	Normal nucleoli	1–10	2.87	3.05
9	Mitoses	1–10	1.60	1.73

(1996) used 10-fold cross-validation with C4.5 decision tree method and achieved a classification accuracy of 94.74%. Hamilton, Shan, and Cercone (1996) used RIAC method to achieve the accuracy of 94.99%. Nauck and Kruse (1999) used neuron-fuzzy techniques to obtain the accuracy of 95.06%. Pena-Reyes and Sipper (1999) used the fuzzy-GA method and reached a classification accuracy of 97.36%. Albrecht, Lappas, Vinterbo, Wong, and Ohno-Machado (2002) used a combination of perceptron algorithm with simulated annealing and reported accuracy of 98.8%. Abonyi and Szeifert (2003) used supervised fuzzy clustering technique to obtained an accuracy of 95.57%. Polat and Güneş (2007) used artificial immune recognition system (AIRS) and fuzzy resource allocation mechanism to obtain an accuracy of 98.51%. Übeyli (2007) five different classifiers, support vector machine, probabilistic neural network, recurrent neural network, combined neural network and Multilayer Perceptron neural networks, were applied and respective accuracies of 99.54%, 98.61%, 98.15%, 97.40% and 91.92% was obtained. Least Square SVM (LS-SVM) was used by Polat and Güneş (2007) to obtained 98.53% accuracy. Peng, Wu, and Jiang (2010) applied integration of wrapper and filter approaches to obtain an accuracy of 99.5%. Örkcü and Bal (2011) compared the performance of Back Propagation Neural Network (BPNN), Binary coded Genetic Algorithm and Real Coded Genetic Algorithm on the breast cancer database and achieved an accuracy of 93.1%, 94%, and 96.5% respectively. Marcano-Cedeño et al. (2011) presented a new Artificial Metaplasticity Multilayer Perceptron algorithm which performed better than BPNN on the same breast cancer database to give the classification accuracy of 99.26% as compared to BPNN 94.51% for 60/40 training–testing samples. Lavanya and Rani (2011) used decision tree algorithms for the same and achieved 92.97% classification accuracy. Malmir et al. (2013) achieved an accuracy of 97.75% and 97.63% by training a Multilayer Perceptron (MLP) Network for 40 iterations using Imperialist Competitive Algorithm (ICA) and Particle Swarm Optimization (PSO) respectively. Koyuncu and Ceylan (2013) achieved a higher classification accuracy of 98.05% by using 9 classifiers in a Rotation Forest-Artificial Neural Network (RF-ANN). Xue, Zhang, and Browne (2014) presented a Particle swarm optimization technique for feature selection using novel initialization and updating mechanisms PSO (4–2) to obtained an accuracy of 94.74%.

In this study, a Genetically Optimized Neural Network (GONN) model is proposed which simultaneously evolved the structure and weight of neural network for classifying the WBCD breast cancer database as benign or malignant. Our algorithm is inspired by Koza and Rice (1991), to optimize the structure of a neural network by genetic evolution. GONN implements new crossover and mutation operators in GP to eliminate the destructive nature of crossover and mutation operations in a standard GP life-cycle (Koza, 1992). To measure the performance of the proposed algorithm we used the Wisconsin breast cancer dataset from the UCI Machine Learning Repository (Bache & Lichman, 2013). It is observed that the proposed algorithm yielded an accuracy of 98.24%, 99.63%, 100%, for 50–50, 60–40, 70–30 and training–testing partition respectively and classification accuracy of 100% for 10-fold cross validation scheme. Measures such as sensitivity, specificity, ROC curves, Area under the ROC curves (AUC) and Mann–Whitney two tailed test are used to validate the performance. To show the dominance of our approach, we compared our method with a classical Koza and Rice (1991) model, classical Back Propagation Neural Network (BPNN) (Hagan, Martin, & Beale, 1996) and also with recently proposed algorithms applied on the WBCD database. The results show that our approach works well with the breast cancer database and can be a good alternative to the well-known machine learning methods.

2. Wisconsin breast cancer database description

In this study, we had performed our experiment on WBCD database taken from UCI Machine Learning repository (Bache & Lichman, 2013). The Wisconsin Breast Cancer database consists of 699 instances taken from Fine Needle Aspirates (FNA) of human breast tissue. The dataset has 9 attributes and its class (benign or malignant) corresponding to each record. The value of each attribute listed in Table 1 is an integer value between 1 and 10, the value of 10 indicates the most abnormal state. Out of the 699 instances, 16 of them have some missing attribute values. So, we discard them in our experimentation and consider only the remaining 683 samples. 444 of them belong to benign class and 239 to malignant class. The goal of our algorithm would be to correctly classify the samples as benign or malignant because a tumor can be benign (not harmful) or malignant (has the potential to be harmful).

3. Brief review of Genetic Programming and Artificial Neural Networks

3.1. Genetic Programming

Genetic Programming (GP) (Koza, 1992), an evolutionary machine learning approach is inspired by Darwin's theory of evolution. It initially generates random solutions to solve a problem, and then evolves them based on a fitness function. New and improved individuals are produced by applying reproduction, crossover and mutation operators on individuals of previous generation. Reproduction is an asexual method where in a selected individual copies itself into the new population. It is effectively the same as one individual surviving into the next generation. Crossover is applied in a GP by simply exchanging sub-trees between two trees, thus forming two new offspring from two parents. Mutation changes a node within a tree or changes its information, thus, affecting only the individual and creating a new solution. Due to crossover and mutation operators, GP techniques do a better job in exploring the search space than other machine learning algorithms. Thus make the problem to converge fast with global optima. It is widely used by many researchers for solving optimization problems (Baydar & Saitou, 2001; Barmpalexis, Kachrimanis, Tsakonas, & Georgarakis, 2011; Pérez, Cladera, Rabuñal, & Martínez-Abella, 2012; Mousavi, Esfahanipour, & Zarandi, 2014). Although GP shows good performance, but still there is a need to improve the crossover and mutation operators as they produce individuals which are poor in fitness as compared to parents (Muni, Pal, & Das, 2004), due to which GP takes more time to reach the desired solution.

3.2. Artificial Neural Network

Artificial Neural Networks (ANN) (Hopfield, 1988), a very popular machine learning technique, inspired by the human brain containing a biological neural network. It consists of good adaptation/learning ability. Feed-forward neural network (Bebis & Georgiopoulos, 1994) are very common type of ANN, in which each artificial neuron has weights assigned to it, with its inputs coming from neurons in the previous layer, and output is passed to the next layer after processing. An important class of feed-forward neural network are Multilayer Perceptron (MLP) (Haykin, 2008). Back-propagation algorithm is the most widely used MLP training technique, which iteratively changes the weights between the neurons in a direction that minimizes the error. This model is very good at learning patterns. It can easily adapt to new trends in data but the problem with this is its slow rate of convergence and the risk of being trapped in a local optimum (Rumelhart, Hinton, & Williams, 1985).

Another problem with BPA is in deciding its structure, the number of layers and number of hidden neurons in each layer and the connectivity between them. It plays a very crucial role in the performance of neural network. The results can vary largely with a slight variation in any of these parameters. Different ANN architectures will give different results for different problems. Application problems will always require a structure that will give near optimal results. However, coming to an optimal ANN architecture through trial-and-error is a mammoth task.

Koza and Rice (1991) brought a new dimension in this area, by using genetic evolution to optimize both, the weight and structure of a neural network. There have been many other works in this area (Palmes, Hayasaka, & Usui, 2005; Rivero, Dorado, Rabuñal, & Pazos, 2010; Tsai & Lin, 2011; Turner, Dudek, & Ritchie, 2010; Mahsal Khan, Masood Ahmad, Muhammad Khan, & Miller, 2013) however; none of the modifications is capable of delivering satisfactory performance for all problems, in general. Thus, the search for an approach to speed up its convergence and find the optimal structure still remain important. For this reason, we present a novel Genetically Optimized Neural Network Model which finds the optimal structure and weights of neural network architecture. In this, we propose a modified crossover and mutation operators which expand the search area and enhance the performance. In the next section, we will describe the propose algorithm.

4. Proposed Genetically Optimized Neural Network (GONN)

To evolve the ANN architecture using GP, for solving the classification of WBCD dataset, we have to form a GONN architecture in such a way to treat it like an ANN structure. To build the GONN architecture, we have to follow the steps of GP life cycle with modified crossover and mutation operators. Thus a final GONN architecture represents an ANN with appropriate mapping of GP parameters to ANN learning parameters. The detail description of initialization method, fitness function, proposed crossover and mutation operators, termination criteria, mapping of GONN architecture to its equivalent Feed Forward Neural Network representation are described next:

4.1. Initialization

GP evolves computer functions, traditionally represented in memory as tree structures. Every internal tree node has an operator function and every terminal node has an operand, making mathematical expressions easy to evolve and evaluate. We restrict the tree generation in GP in such a way that it preserves neural network architecture, so that we can convert the GP based classifier to an ANN. For achieving this, let a possible function set F and Terminal set T for a GP tree are,

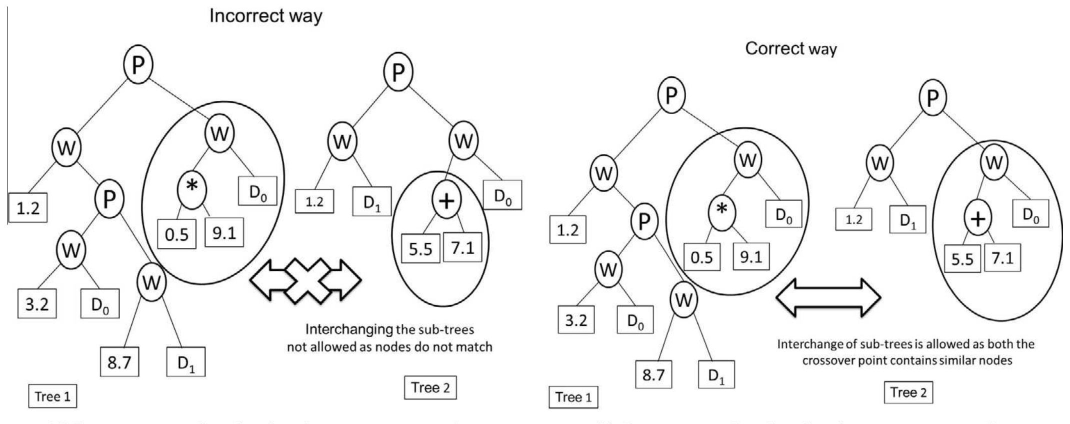
$$F = \{P, W, +, -, *, \%\} \tag{1}$$

$$T = \{feature\ variables\ of\ dataset, R\} \tag{2}$$

where, P is an activation function which is standard sigmoid function in our case whose range is [0,1].

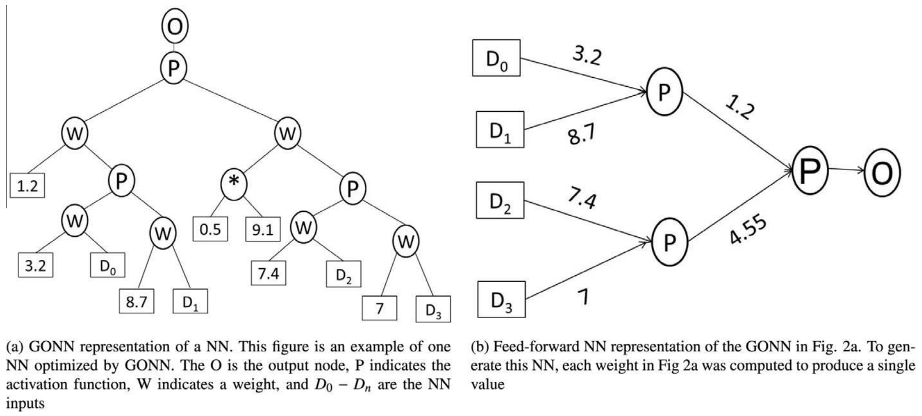
W is the weighting function for a signal going into P.
$+, -, *$ and % are normal arithmetic functions.
$D_0, D_1, \ldots D_n$ are the feature variables (input data signal) of dataset.
R contains randomly generated floating point constants between 0.0 and 10.0, because the input variables in our dataset have values in the range of 1–10.

The structure consists of an activation function (the P function) that process weighted inputs to produce a discrete signal (i.e. 0 or


Incorrect way
Interchanging the sub-trees not allowed as nodes do not match
Tree 1
Tree 2
Correct way
Interchange of sub-trees is allowed as both the crossover point contains similar nodes
Tree 1
Tree 2

(a) Incorrect way of performing the crossover operation (b) Correct way of performing the crossover operation

Fig. 1. Showing the way to perform the crossover in GONN architecture so that it preserves the neural network architecture.



(a) GONN representation of a NN. This figure is an example of one NN optimized by GONN. The O is the output node, P indicates the activation function, W indicates a weight, and $D_0 - D_n$ are the NN inputs

(b) Feed-forward NN representation of the GONN in Fig. 2a. To generate this NN, each weight in Fig 2a was computed to produce a single value

Fig. 2. Conversion of GONN architecture to its equivalent Feed Forward Neural Network architecture.

1) as its output. Here, the standard sigmoid function is used as an activation function whose range is [0,1] and is written as follows:

$$y = \frac{1}{1 + e^{-x}} \tag{3}$$

where $x \in [-1, 1]$. P adds up its two weighted input lines and emits a 1 if the sum exceeds the value of 0.5 and emits a 0 otherwise. The weighting function W is, in fact, merely multiplication, given a special name to distinguish it from the ordinary arithmetic operation of multiplication. Both P and W can have varying number of arguments. The four arithmetic operations are used to create and modify the numerical constants (weights) of the neural network. In order to evolve ANN with GP, we have to follow some rules in applying the function set and terminal set to form a GP tree. The rules of construction for a neural network with one output signal, which are laid down by Koza and Rice (1991) are as follows:

(1) The root of the tree must be an activation function P.
(2) The only thing allowed at the level immediately below any activation function P is a weighting function W.
(3) The only thing allowed below a weighting function W on the left child is either a floating-point random constant or an arithmetic function.
(4) The only thing allowed below a weighting function W on the right child is either an input data signal (such as D_0 and D_1) or the output of a P function.
(5) The only thing allowed below an arithmetic function is either a floating-point random constant or an arithmetic function $(+, -, *, \%)$.

Note that the external points of the tree are either input signals (i.e. D_0, D_1, etc.) or floating-point random constants. These rules are applied recursively while generating GP trees.

4.2. Fitness function

The most challenging and yet an essential concept of GP is the fitness function. The fitness function determines the ability of a program to solve the problem. Fundamentally, GP is guided by a fitness function, which explores for an efficient program to solve a given problem. It helps in determining the possible solutions for evolving into next generation of solutions.

The fitness evaluation is done by simply obtaining the mean-squared error of the GONN output.

$$Fitness_{GONN} = \frac{1}{1 + \frac{1}{X}\sum_{Y}^{X}(DES_Y - ACT_Y)^2} \tag{4}$$

here X is the size of the entire training dataset, DES_Y is the desired output from using the training dataset number Y, and ACT_Y is the GONN output when using the training dataset number Y as the input.

However, we bring some changes in the GP life-cycle, which deviates from the normal approach in GP which are presented next:

4.3. Modified Crossover Operator

After applying the reproduction operator, in which we select the top P_r% of top fitness individuals and transfer them to next generation, a typical GP life-cycle would take a subset of the remaining individuals for crossover operation and the rest for mutation operation. However, we consider all the individuals left after reproduction for crossover operation. Also, our crossover operator has to be consistent with tree structure rules. A subtree with root node W or P can be swapped only with another subtree having the same functional node i.e. W or P respectively as shown in the Fig. 1. During crossover, we give 90% chance to functional nodes and 10% to terminal nodes for swapping. The newly generated offspring are then evaluated for their fitness. An offspring is allowed to enter the next generation only if it is better than its parent, in terms of its fitness value. In other cases, the crossover operation is applied again until the performance of offspring improves. All the generated offspring are kept in a sorted order in a table. A certain top percentage (P_c%) of the new individuals are then selected to be carried forward to the next generation. The parent trees (P_m%) which did not generate good fitness offspring are selected for mutation operation.

There are advantages of these modifications over normal crossover operation. Since all individuals are first considered for crossover, the variety of offspring individuals increases. The 90% bias rule causes a bigger movement in the search space, which helps in reaching the solution faster. By allowing offspring into the next generation only if they are better than their parents, eliminates the destructive nature of crossover.

4.4. Modified mutation operator

The individuals which are not good for producing better offspring than themselves are changed in our mutation and we bring the diversity in them very early. However, the bias in crossover brings in a disadvantage that in later GP generations, it becomes more difficult to fine tune candidate solutions by changing a leaf of the tree. To overcome this problem, during mutation operation, we give 90% bias to terminal nodes and 10% to functional nodes for swapping. If the fitness value of child is less than the parent, the solution is discarded and mutation function is applied again on the same parent, until it generates better child than himself. Thus, making the crossover and mutation operators more constructive.

4.4.1. Termination criteria

In this GONN approach, the learning/evolutionary process is terminated, if it meets with any of the below mentioned conditions.

1. If the number of fitness evaluations reaches its maximum count.
2. If the mean square error reaches 0.01.

Through evolution, the newly generated individuals are better than their previous population. The best individual in terms of fitness value is considered as the optimal GONN architecture. The step-wise procedure for evolving GONN architecture is described in the Algorithm 1. After that, mapping of GONN tree is done to its equivalent Feed Forward Neural Network, which is then used for testing data. Fig. 2a shows an example of a tree representation of a NN generated by GONN. Fig. 2b shows the same NN that has been reduced from the GONN tree. It contains one input layer with four inputs D_0, D_1, D_2, D_3, one hidden layer with two neurons and one output layer.

Algorithm 1. Algorithm for Evolving Genetically Optimized Neural Network (GONN) architecture

```
1: **Input** WBCD training data and GP parameters see Table 2.
2: **Output** Genetically Optimized Neural Network (GONN) architecture.
3: **Begin**
4: **Initial Population** Generate initial classifier population of size $k$ with input data.
5: **while** No. of fitness evaluations < Maximum number of fitness evaluations. **do**
6:     **Fitness Evaluation** Calculate the fitness value of all individuals.
7:     **Reproduction** Select top $P_r$% of individuals to be carried forward to the next generation.
8:     **Crossover** Apply modified crossover operation on all the remaining parent pairs.
9:     **for all** crossover pairs **do**
10:        **Repeat till we get offspring better than parents** *Take the parent pair and generate two offspring from them by giving more chances to function nodes (90%) to swap.*
11:        **Sort and store** *Place the top offspring of that pair into a table sorted according to fitness values.*
12:    **end for**
13:    **Selection** Select the top $P_c$% offspring from the sorted table and transfer them to the next generation.
14:    **Mutation** Take the parents of remaining $P_m$% individuals in the table and apply modified mutation.
15:    **for all** mutation parents **do**
16:        **Repeat till we get offspring better than parents** *Take the parent and generate child from them by giving more chances to terminal nodes (90%) to mutate.*
17:    **end for**
18:    **Selection** *Transfer these $P_m$% offspring to the next generation.*
18: **end while**
20: **return** *Best individual in terms of fitness value which is considered as Genetically Optimized Neural Network (GONN) architecture.*
21: **End**
```

4.5. Mapping of a GONN architecture to its equivalent Feed Forward Neural Network

During training phase, we evolve the Feed Forward Neural Network using GONN architecture. After getting the optimal GONN architecture, we convert it into an equivalent Feed Forward Neural Network. For converting the optimal GONN architecture to equivalent Feed Forward Neural Network, we followed the following rules:

1. Left child of weight function (W) will become the weight value for the activation function (from which the weight function is connected in GONN tree).
2. If the left child of W is arithmetic function than output of that arithmetic function will become the weight value of the activation function (from which the weight function is connected in GONN tree).
3. Right child of weight function (W) will become the input for the activation function (from which the weight function is connected in GONN tree).
4. If the right child of W is an activation function P, then P becomes the neuron of a hidden layer of neural network.

Now this Feed Forward Neural Network representation of GONN architecture is used for testing WBCD dataset. The Algorithm 2 represents the Feed Forward Neural Network generated by GONN architecture applied on testing WBCD dataset. In this, we calculate the output value of each neuron (net) by summing the product of inputs and weights layer by layer and applying the sigmoid activation function. If the output of net is greater than 0.5 the value of net becomes 1 otherwise 0. The output of first layer becomes the input of second layer and in this way the final output of the architecture ($Output_{GONN}$) is evaluated. If the final output of the GONN structure is 1 than sample belongs to class 1 (Malignant) otherwise to class 0 (Benign).

Algorithm 2. Algorithm for testing Feed Forward Neural Network generated by GONN

```
1: **Input** WBCD testing data and optimal Feed Forward Neural Network generated by GONN model.
2: **Output** Classification of testing samples.
3: **Begin**
4: **for all** test samples **do**
5:     **for all** $i$ = 1 to m layers **do**
6:         **for all** $j$ = 1 to n neurons in each layer **do**
7:             $net_j^i = \sum W_j^{i-1} * D_j^{i-1}$
8:             **if** $f(net_j^i) \geqslant 0.5$ **then**
9:                 set $net_j^i = 1$
10:            **else**
11:                set $net_j^i = 0$
12:            **end if**
13:            $D_j^i = net_j^i$
14:        **end for**
15:    **end for**
16:    **if** $Output_{GONN} = 1$ **then**
17:        test sample $\in$ **class 1 (Malignant)**
18:    **else**
19:        test sample $\in$ **class 0 (Benign)**
20:    **end if**
21: **end for**
22: **End**
```

Table 2
Parameters value for GONN and Koza and Rice (1991) models.

Parameters	Value
Probability of crossover operation (P_c)	60%
Probability of reproduction operation (P_r)	20%
Probability of mutation operation (P_m)	20%
Population size (k)	100
Initialization method	Ramped half and half
Initial maximum depth of tree	6
Initial minimum depth of tree	3
Function Set	+, −, ∗, / (protected division, division by zero is zero)
Terminals	Feature variables from datasets, floating point constants [0.0,10.0]
Termination criteria	40,000 Fitness evaluation or MSE = 0.01

Table 3
Parameters for classical back propagation algorithm.

Parameters	Value
Number of input neurons	10 (9 attribute values and 1 1 bias input)
Number of hidden layers	1
Number of hidden neurons	8
Number of output neurons	1
Learning rate	0.1
Momentum constant	0.9
Termination criteria	MSE = 0.01 or Total number of epochs = 2000

5. Results and discussion

The proposed GONN as a classifier was implemented in Java (Java SE 6 Update 45) and on a Pentium IV computer of 3.4 GHz with 2 GB of RAM. This algorithm was applied to the Wisconsin breast cancer Database (WBCD). Experimentation is carried out on the dataset with the parameters as described in Table 2.

In machine learning field, it is common to partition the dataset into two separate sets: a training set and a testing set. To evaluate the generalizability of our approach and to compare our work with existing work in literature, we divided the training and testing data into four different partitions. A standard 50–50 methodology was used, where half of the samples are used for training the classifier, and the rest to testing. We also divided the dataset into 60–40 and 70–30 training–testing ratios to show the importance of training data for our proposed approach. Here, a 10-fold cross validation technique (Hastie, Friedman, & Tibshirani, 2009) was also used to calculate classification accuracy of our model. In this method, entire dataset is divided into ten blocks of approximately equal size. While implementing our algorithm, we use 90% of data to train our model, and the rest 10% for testing. This process is repeated 10 times, with a different data block left out for testing every time. So, total of 100 GP runs are evaluated. For 50–50, 60–40, 70–30 methodology also, 100 GP runs are evaluated. The details of training–testing partition are represented in Table 4.

In case of GONN, we got network structure which contains one input layer, one hidden layer and one output layer (1–1–1) for 50 GP evaluations out of 100, which contains 2 hidden neurons. So, we selected 50 optimal architecture of GONN and Koza's which contains 1-1-1 network structure and fix the network architecture of BPNN to (1-1-1) in order to present the fair comparison between all the methods. So, the network structure containing one input layer, one hidden layer and one output layer (1-1-1) are used to compare the performance of GONN, BPNN and Koza's models. In case of GONN and Koza's architecture, only four inputs in input layer are required as GONN and Koza's models are capable of selecting only the important features from the dataset but BPNN requires all the features in the input. For this reason, the BPNN architecture contains 10 inputs in the input layer (9 attributes of dataset and 1 bias input) and 8 neurons in hidden layer. The parameters value for GONN and Koza's models are presented in Table 2 and the parameters value for classical back propagation model is presented in Table 3. The learning rate and momentum constant are chosen as 0.1 and 0.9, respectively for BPNN architecture. The activation function for all the algorithms is the standard sigmoid and it is same for all the neurons. Two different criterion were applied to stop the training: in one case it was stopped when the mean square error reached 0.01, and in the other the training was stopped when total number of fitness evaluation reached 40,000 for GONN and Koza's, and total number of epochs reached 2000, in case of BPNN model. The number of epochs, is the number of times the training samples were presented to the network. The maximum, mean and standard deviation of the results are presented for 50 repetitions of all the algorithms for 50–50, 60–40, 70–30 training–testing partitions and for 10-fold cross validation scheme.

5.1. Performance evaluation methods

In order to compare and evaluate the performance of our proposed GONN architecture, we define and compute the classification accuracy, sensitivity, specificity, confusion matrix, ROC curves and Area under the ROC curves (AUC). We also performed a statistical validation of the results with the Mann–Whitney (Corder & Foreman, 2009) two tailed test, to show the statistical difference in results. The formulations are as follows:

5.1.1. Accuracy

The measure of the ability of the classifier to produce accurate diagnosis is determined by accuracy.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \times 100 \tag{5}$$

5.1.2. Sensitivity

The measure of the ability of the model to identify the occurrence of a target class accurately is determined by sensitivity.

$$Sensitivity = \frac{TP}{TP + FN} \times 100 \tag{6}$$

5.1.3. Specificity

The measure of the ability of the model to separate the target class is determined by specificity.

$$Specificity = \frac{TN}{TN + FP} \times 100 \tag{7}$$

where TP, TN, FP, and FN denote true positives, true negatives, false positives, and false negatives, respectively.

- True positive (TP): An input is detected as a patient with breast cancer, as diagnosed by the expert clinicians.
- True negative (TN): An input is detected as normal and also labeled as a healthy person by the expert clinicians.
- False positive (FP): An input is detected as a patient with breast cancer, although labeled as a healthy person by the expert clinicians.
- False negative (FN): An input is detected as normal, although diagnosed by the expert clinicians as having breast cancer.

Table 4
Training and testing set partition.

Training–testing partition (%)	No. of records in the subset					
	Total training records	Benign records in training set	Malignant records in training set	Total testing records	Benign records in testing set	Malignant records in testing set
50–50	341	222	119	342	222	120
60–40	410	266	144	273	178	95
70–30	478	311	167	205	133	72
10-fold cross validation	615	400	215	68	44	24

Table 5
Confusion matrix.

	Predicted positive	Predicted negative
Actual positive	True positive (TP)	False negative (FN)
Actual negative	False positive (FP)	True negative (TN)

5.1.4. Confusion matrix

A confusion matrix contains information about actual and predicted classifications performed by a classifier. Performance of such classifiers is commonly evaluated using the data in the matrix. A confusion matrix for two class problem is given in Table 5.

5.1.5. ROC Curve

Receiver Operating Characteristic Curve is a fundamental tool for diagnostic test evaluation. It is a two dimensional plot of true positive fraction vs. false positive fraction (sensitivity v/s 1-specificity). Each point on the ROC curve represents a sensitivity/specificity pair corresponding to a particular decision threshold. A good test is one for which sensitivity rises rapidly and 1-specificity hardly increases at all until sensitivity become high. The area under the ROC curve (AUC) is a measure of how well a parameter can distinguish between two diagnostic groups (diseased/normal). AUC lies in the range of $[0,1]$. AUC value close to 1 indicates very reliable diagnostic test.

5.1.6. Mann–Whitney Test

Mann–Whitney (Corder & Foreman, 2009) two tailed test is used to show the statistical significance of the results. In this test, the p values are obtained by comparing the classification accuracy of methods. The difference between the two samples is highly significant if the value of $p < 0.001$ and for the value of $p > 0.05$ the results are not significantly different.

5.2. Experimental results

In this section, we present the experimental results to test the behavior of the proposed GONN method, as well as to compare it with classical BPNN and Koza's models. The results of classification accuracy of all models are presented in Table 6 with different training and testing data. It is clear from the results that GONN model outperforms the classical BPNN and Koza's models in terms of classification accuracy for all training–testing data. This improvement in classification accuracy is mainly due to our modified crossover and mutation operators. Classification accuracy achieved by GONN model for 50–50 training–testing samples is 98.24% which is very remarkable, it shows that even for smaller training samples, GONN model is having good adaptation and generation capability and is able to provide better solutions. The classification accuracy of GONN model for 60–40 and 70–30 are 99.63% and 100% respectively which confirms the importance of proper training data. BPNN and Koza's models also shows improvement in classification accuracy when the training data increases but it is not equivalent to GONN. The classification accuracy for 10-fold cross validation method is 100% which is much superior in comparison to BPNN and Koza's methods. Still in BPNN, convergence to global optimum is always a problem. In comparison, our approach guarantees global optimum with better generalized solutions. Our results confirm that our GONN model can be a very helpful tool to assist the physicians to diagnose the patient or it can be used as a second opinion for their final diagnosis.

The minimum, maximum and mean values of sensitivity and specificity of all training–testing partitions for BPNN, Koza's and GONN models are presented in Table 7. The ROC curves of BPNN, Koza's and GONN models for 50–50, 60–40, 70–30 training–testing partition and 10-fold cross validation scheme are shown in Fig. 3a, b, c and d respectively. The area under the ROC curves (AUC) is computed using the trapezoidal rule (by integrating the areas of trapezoids under the ROC curve). The AUC of GONN model were 0.978, 0.989, 0.998 and 1.0 for 50–50, 60–40, 70–30 training–testing partition and 10-fold cross validation scheme respectively. This AUC of GONN model were much bigger than BPNN and Koza's for every training–testing partition, which once again confirms the superiority of GONN over classical BPNN and Koza's models.

Tables 8 and 9 represents the classification accuracies in the form of confusion matrix. It is clear from Tables 8 and 9, the sum of true positive and true negative increases with the increase in training set size. Especially, for 70–30% and 10-fold cross validation training–testing data the true positive rate and true negative rate is 100% which again shows that, if proper training data is provided our GONN model is having good generalization and adaptation capability.

The statistical difference in result is also presented using the Mann Whitney two tailed test in Table 10. It is clear from the results that the solution produce by our GONN model are statistically different from the BPNN and Koza's models for every training–testing partition.

Table 6
Comparison of classification accuracies of BPNN, Koza and Rice (1991) and GONN classifiers with different validation techniques for 1-1-1 network architecture.

Training–testing partition	BPNN			Koza and Rice (1991)			**GONN**		
	Mean	**Std dev**	**Max**	**Mean**	**Std dev**	**Max**	**Mean**	**Std dev**	**Max**
50–50	89.81	0.288	90.05	89.63	0.930	90.64	97.73	0.476	98.24
60–40	92.28	0.370	92.67	92.84	1.103	93.77	99.11	0.5581	99.63
70–30	93.45	0.740	94.14	94.14	0.232	94.6	99.21	0.421	100
10-fold cross validation	89.11	1.196	89.7	93.47	1.222	94.6	99.26	0.602	100

Table 7
Comparison of sensitivity and specificity of BPNN, Koza and Rice (1991) and GONN classifiers with different training–testing partition for 1-1-1 network architecture.

Training–testing partition	BPNN				Koza and Rice (1991)				GONN			
	Sensitivity		Specificity		Sensitivity		Specificity		Sensitivity		Specificity	
	Mean	Std dev	Mean	Std dev	Mean	Std dev	Mean	Std dev	Mean	Std dev	Mean	Std dev
50–50	93.9	0.213	82.05	0.426	93.79	1.094	82.98	0.416	98.85	0.228	95.77	0.842
60–40	96.30	0.282	84.77	0.532	96.31	0.858	85.84	1.605	99.17	0.508	98.45	0.532
70–30	96.56	0.378	87.21	1.380	96.26	1.079	87.67	1.349	99.51	0.372	99.21	0.700
10-fold cross validation	94.46	1.144	77.21	2.110	96.5	1.141	88.06	0.655	98.77	1.167	100	0

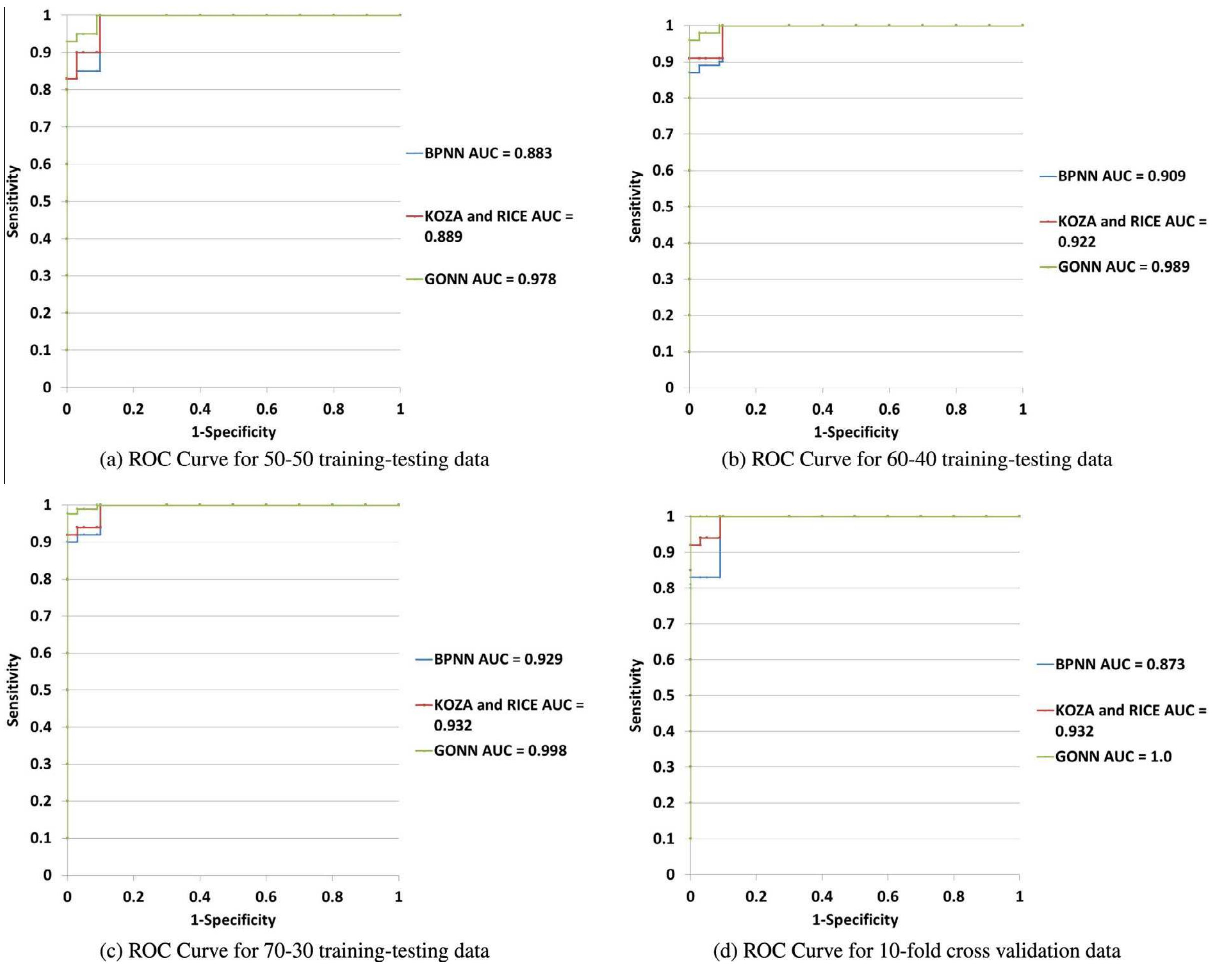


Fig. 3. ROC Curves and AUC of BPNN, Koza's and GONN model for different training–testing data.

Table 8
Confusion matrices for classifiers used for classification of breast cancer for 50–50 and 60–40 training–testing data.

Type classifiers	Desired result	Output results for 50–50 training–testing partition		Output results for 60–40 training–testing partition	
		Benign	Malignant	Benign	Malignant
BPNN	Benign records	209	13	172	6
	Malignant records	21	99	14	81
Koza and Rice (1991)	Benign records	210	12	173	5
	Malignant records	20	100	12	83
GONNs	Benign records	219	3	178	0
	Malignant records	6	114	1	94

Table 9
Confusion matrices for classifiers used for classification of breast cancer for 70–30 training–testing and 10-fold cross validation data.

Type classifiers	Desired result	Output results for 70–30 training–testing partition		Output results for 10 fold cross validation	
		Benign	Malignant	Benign	Malignant
BPNN	Benign records	129	4	43	1
	Malignant records	8	64	4	20
Koza and Rice (1991)	Benign records	130	3	43	1
	Malignant records	8	64	3	21
GONNs	Benign records	133	0	44	0
	Malignant records	0	72	0	24

Table 10
p-value comparison using Mann Whitney Test.

	Training–testing partition	BPNN		Koza and Rice (1991)	
		p-value	Significance	*p*-value	Significance
GONN	50–50	2.871×10^{-11}	Highly significant	1.9703×10^{-11}	Highly significant
	60–40	2.871×10^{-11}	Highly significant	1.9703×10^{-11}	Highly significant
	70–30	2.871×10^{-11}	Highly significant	1.9703×10^{-11}	Highly significant
	10-fold cross validation	2.871×10^{-11}	Highly significant	1.9703×10^{-11}	Highly significant

Table 11
Performance Comparison with other work from literature.

Author year	Method	Classification accuracy (%)
Quinlan (1996)	C4.5	94.74
Hamilton et al. (1996)	RAIC	95.00
Nauck and Kruse (1999)	Neuro-fuzzy	95.06
Pena-Reyes and Sipper (1999)	Fuzzy-GA	97.36
Albrecht et al. (2002)	LSA machine	98.80
Abonyi and Szeifert (2003)	Supervised fuzzy clustering	95.57
Polat and Güneş (2007)	Fuzzy-AIRS	98.51
Übeyli (2007)	SVM	99.54
Polat and Güneş (2007)	LS-SVM	98.53
Peng et al. (2010)	CFW	99.50
Örkcü and Bal (2011)	Real coded GA	96.5
Marcano-Cedeño et al. (2011)	AMMLP	99.26
Lavanya and Rani (2011)	Decision tree algorithms	92.97
Malmir et al. (2013)	Imperialist Competitive Algorithm (ICA)	97.75
Koyuncu and Ceylan (2013)	RF-ANN	98.05
Xue et al. (2014)	PSO(4–2)	94.74
This study	*GONN*[1]	98.24[a] –97.73[b]
This study	*GONN*[2]	99.63[a] – 99.11[b]
This study	*GONN*[3]	100[a] – 99.21[b]
This study	*GONN*[4]	100[a] –99.26[b]

[a] Best result obtained from one GP run.
[b] Average result obtained from 50 GP runs.
[1] Result of GONN for 50–50 training–testing data.
[2] Result of GONN for 60–40 training–testing data.
[3] Result of GONN for 70–30 training–testing data.
[4] Result of GONN for 10-fold cross validation scheme.

5.3. Comparison with other methods

Proposed method is compared with other work present in literature applied on the Wisconsin breast cancer database. It is clear from Table 11 that our approach, in which we evolve the structure of a neural network genetically, is significantly better in terms of classification accuracy than other approaches. It is important to highlight that, the authors of these work do not specify whether the results provided by them are the maximum accuracy achieved by their classifier or the average accuracy of several runs or in what training–testing partition there data is divided. Proposed method achieve maximum accuracy of 98.24%, 99.63% and100% for 50–50, 60–40 and 70–30 training–testing partition respectively and 100% for 10-fold cross validation and average accuracy of 97.73%, 99.11% and 99.21% for 50–50, 60–40 and 70–30 training–testing partition respectively and 99.26% for 10-fold cross validation for 50 GP runs for 1–1-1 network architecture.

6. Conclusion

In this work, a novel approach for breast cancer diagnosis is explored by an Artificial Neural Network which is genetically evolved to an optimal architecture (structure and weight) for classification. This is done by using the concept of Genetic Programming with proposed crossover and mutation operators in which the destructive nature of these operators is eliminated. Also, the suggested changes bring more diversity in the GP population and help the algorithm to reach solution faster with more generalized solutions. The proposed method is experimented and compared with the classical BPNN and Koza's models applied to the WBCD database. Proposed method achieves maximum accuracy of 98.24%, 99.63% and 100% for 50–50, 60–40, and 70–30 training–testing partitions respectively and 100% for10 fold cross validation. An average accuracy of 97.73%, 99.11% and 99.21% is achieved for 50–50, 60–40 and 70–30 training–testing partitions respectively and 99.26% for 10-fold cross validation scheme. The results of the confusion matrix shows that with increase number of training samples, the number of false positive and false negative rates decreases. The AUC values for our method are 0.978, 0.989 and 0.998 for 50–50, 60–40 and 70–30 training–testing partitions respectively and 1.0 for 10-fold cross validation. These results are much better than BPNN and Koza's models. From the above results, we conclude that our proposed GONN model obtains very high accuracy in classifying the WBCD breast cancer data. We believe that the GONN model can be a very helpful tool to assist the physicians to diagnose the patient or it can be used as a second opinion for their final diagnosis. Our GONN model shows the competent or better results when compared with the state-of-the art methods applied to the WBCD database.

In the aspects concerning the limitation of this research, it is important to notice that only crossover and mutation operators are improved. There are other areas like feature extraction, feature selection which help to extract the features even from images and various signals used during medical diagnosis and reduce the irrelevant and redundant features to make GONN more efficient for real time diagnosis of breast cancer. Other point that can be cited here is that the GONN model is applied on WBCD dataset available on UCI repository which is very small. Thus, generalizability of the proposed model is further tested with variety of actual data from different hospitals.

There are few aspects of this research that could be improved further or extended in nearest future. The proposed GONN model is proposed to apply on breast cancer diagnosis which is a 2-class problem; it would be interesting to see its behavior on other multi-class classification problems. Moreover, in future the research can be carried out for screening of features that can reduce the computational burden of the algorithm with further improvement in the classification accuracy to diagnose breast cancer tumor. The proposed model is applied on numeric data only; it would be interesting to see its behavior when it is applied on different types of data available in medical field such as images, signals. However, for practical implementation, future work is required to evaluate the usefulness of the proposed method with more number of patients.

References

Abonyi, J., & Szeifert, F. (2003). Supervised fuzzy clustering for the identification of fuzzy classifiers. *Pattern Recognition Letters, 24*(14), 2195–2207.

Albrecht, A. A., Lappas, G., Vinterbo, S. A., Wong, C., & Ohno-Machado, L. (2002). Two applications of the lsa machine. *Neural Information Processing, 2002. Proceedings of the 9th International Conference on ICONIP'02* (vol. 1, pp. 184–189). IEEE.

Bache, K., & Lichman, M., 2013 UCI machine learning repository. URL: http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/.

Barmpalexis, P., Kachrimanis, K., Tsakonas, A., & Georgarakis, E. (2011). Symbolic regression via genetic programming in the optimization of a controlled release pharmaceutical formulation. *Chemometrics and Intelligent Laboratory Systems, 107*(1), 75–82.

Baydar, C. M., & Saitou, K. (2001). Automated generation of robust error recovery logic in assembly systems using genetic programming. *Journal of Manufacturing Systems, 20*(1), 55–68.

Bebis, G., & Georgiopoulos, M. (1994). Feed-forward neural networks. *IEEE Potentials, 13*(4), 27–31.

Bellazzi, R., & Zupan, B. (2008). Predictive data mining in clinical medicine: current issues and guidelines. *International Jorunal of Medical Informatics, 77*(2), 81–97.

Carbonell, J. G. (1983). *Learning by analogy: Formulating and generalizing plans from past experience*. Springer.

Christoyianni, I., Dermatas, E., & Kokkinakis, G. (2000). Fast detection of masses in computer-aided mammography. *IEEE Signal Processing Magazine, 17*(1), 54–64.

Corder, G. W., & Foreman, D. I. (2009). *Nonparametric statistics for non-statisticians: A step-by-step approach*. John Wiley & Sons.

Hagan, D. H. B., Martin, T., & Beale, M. H. (1996). *Neural network design* (Vol. 1). Boston: Pws.

Hamilton, H. J., Shan, N., & Cercone, N. (1996). RIAC: A rule induction algorithm based on approximate classification. Computer Science Department, University of Regina.

Hastie, T., Friedman, J., & Tibshirani, R. (2009). *The elements of statistical learning* (Vol. 2). Springer.

Haykin, S. S. (2008). *Neural networks and learning machines* (3 edition.). Prentice Hall.

Hopfield, J. J. (1988). Artificial neural networks. *IEEE Circuits Devices Magazine, 4*(5), 3–10.

Koyuncu, H., & Ceylan, R. (2013). Artificial neural network based on rotation forest for biomedical pattern classification. In *36th international conference on telecommunications and signal processing (TSP), 2013* (pp. 581–585). IEEE.

Koza, J. R. (1992). *Genetic programming: On the programming of computers by means of natural selection* (Vol. 1). MIT press.

Koza, J. R., & Rice, J. P. (1991). Genetic generation of both the weights and architecture for a neural network. *IJCNN-91-seattle international joint conference on neural networks, 1991* (vol. 2, pp. 397–404). IEEE.

Lavanya, D., & Rani, D. K. U. (2011). Analysis of feature selection with classification: Breast cancer datasets. *Indian Journal of Computer Science and Engineering (IJCSE), 2*(5), 756–763.

Mahsal Khan, M., Masood Ahmad, A., Muhammad Khan, G., & Miller, J. F. (2013). Fast learning neural networks using cartesian genetic programming. *Neurocomputing, 121*, 274–289.

Maimon, O. Z., & Rokach, L. (2005). *Data mining and knowledge discovery handbook* (Vol. 1). Springer.

Malmir, H., Farokhi, F., & Sabbaghi-Nadooshan, R. (2013). Optimization of data mining with evolutionary algorithms for cloud computing application. In *3th international econference on computer and knowledge engineering (ICCKE), 2013* (pp. 343–347). IEEE.

Marcano-Cedeño, A., Quintanilla-Domínguez, J., & Andina, D. (2011). Wbcd breast cancer database classification applying artificial metaplasticity neural network. *Expert Systems with Application, 38*(8), 9573–9579.

Mitra, S., & Acharya, T. (2005). *Data mining: multimedia, soft computing, and bioinformatics*. John Wiley & Sons.

Mitra, S., & Hayashi, Y. (2000). Neuro-fuzzy rule generation: Survey in soft computing framework. *IEEE Transactions on Neural Networks, 11*(3), 748–768.

Mousavi, S., Esfahanipour, A., & Zarandi, M. H. F. (2014). A novel approach to dynamic portfolio trading system using multitree genetic programming. *Knowledge-Based Systems*.

Muni, D. P., Pal, N. R., & Das, J. (2004). A novel approach to design classifiers using genetic programming. *IEEE Transactions on Evolutionary Computation, 8*(2), 183–196.

Muto, T., Bussey, H., & Morson, B. (1975). The evolution of cancer of the colon and rectum. *Cancer, 36*(6), 2251–2270.

Nauck, D., & Kruse, R. (1999). Obtaining interpretable fuzzy classification rules from medical data. *Artificial Intelligence in Medicine, 16*(2), 149–169.

Örkcü, H. H., & Bal, H. (2011). Comparing performances of backpropagation and genetic algorithms in the data classification. *Expert Systems with Application, 38*(4), 3703–3709.

Palmes, P. P., Hayasaka, T., & Usui, S. (2005). Mutation-based genetic neural network. *IEEE Transactions on Neural Networks, 16*(3), 587–600.

Pena-Reyes, C. A., & Sipper, M. (1999). A fuzzy-genetic approach to breast cancer diagnosis. *Artificial Intelligence in Medicine, 17*(2), 131–155.

Peng, Y., Wu, Z., & Jiang, J. (2010). A novel feature selection approach for biomedical data classification. *Journal of Biomedical Informatics, 43*(1), 15–23.

Pérez, J. L., Cladera, A., Rabuñal, J. R., & Martínez-Abella, F. (2012). Optimization of existing equations using a new genetic programming algorithm: Application to the shear strength of reinforced concrete beams. *Advances in Engineering Software, 50*, 82–96.

Polat, K., & Güneş, S. (2007). Breast cancer diagnosis using least square support vector machine. *Digital Signal Processing, 17*(4), 694–701.

Quinlan, J. R. (1996). Improved use of continuous attributes in c4. 5. *Journal of Artificial Intelligence Research, 4*, 77909.

Rivero, D., Dorado, J., Rabuñal, J., & Pazos, A. (2010). Generation and simplification of artificial neural networks by means of genetic programming. *Neurocomputing, 73*(16), 3200–3223.

Rodrigues, P. S., Giraldi, G. A., Chang, R.-F., & Suri, J. S. (2006). Non-extensive entropy for cad systems of breast cancer images. *Computer graphics and image processing, 2006. 19th Brazilian symposium on SIBGRAPI'06* (pp. 121–128). IEEE.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1985). Learning internal representations by error propagation. Technical report, DTIC Document.

Tsai, H.-C., & Lin, Y.-H. (2011). Modular neural network programming with genetic optimization. *Expert Systems with Applications, 38*(9), 11032–11039.

Turner, S. D., Dudek, S. M., & Ritchie, M. D. (2010). Grammatical evolution of neural networks for discovering epistasis among quantitative trait loci. In *Evolutionary computation, machine learning and data mining in bioinformatics* (pp. 86–97). Springer.

Übeyli, E. D. (2007). Implementing automated diagnostic systems for breast cancer detection. *Expert Systems with Applications, 33*(4), 1054–1062.

Witten, I. H., & Frank, E. (2005). *Data mining: Practical machine learning tools and techniques*. Morgan Kaufmann.

Xue, B., Zhang, M., & Browne, W. N. (2014). Particle swarm optimisation for feature selection in classification: Novel initialisation and updating mechanisms. *Applied Soft Computing, 18*, 261–276.