Feature Selection and Classification using Flexible Neural Tree

Yuehui Chen[1], Ajith Abraham[1,2] and Bo Yang[1,3]

[1] *School of Information science and Engineering, Jinan University, 106 Jiwei Road, 250022 Jinan, P.R. China*
[2] *IITA Professorship Program, School of Computer Science and Engineering, Chung-Ang University, Seoul, Korea*
[3] *State Key Lab. of Advanced Technology for Materials Synthesis and Processing, Wuhan University of Science and Technology, Wuhan, P.R. China*

Abstract

The purpose of this research is to develop effective machine learning or data mining techniques based on Flexible Neural Tree (FNT). Based on the pre-defined instruction/operator sets, a flexible neural tree model can be created and evolved. This framework allows input variables selection, over-layer connections and different activation functions for the various nodes involved. The FNT structure is developed using Genetic Programming (GP) and the parameters are optimized by a Memetic Algorithm (MA). The proposed approach was applied for two real world problems involving designing Intrusion Detection System (IDS) and for breast cancer classification. The IDS data has 41 inputs/features and the breast cancer classification problem has 30 inputs/features. Empirical results indicate that the proposed method is efficient for both input feature selection and improved classification rate.

Key words: Flexible neural tree model, genetic programming, memetic algorithm, intrusion detection system, breast cancer classification.

1 Introduction

Variable selection refers to the problem of selecting input variables that are most predictive for a given outcome. Appropriate variable selection can greatly enhance the effectiveness and potential interpretability of an inference model.

Email address: yhchen@ujn.edu.cn, ajith.abraham@ieee.org, yangbo@ujn.edu.cn (Yuehui Chen[1], Ajith Abraham[1,2] and Bo Yang[1,3]).

Variable selection problems are found in all supervised and unsupervised machine learning tasks including classification, regression, time-series prediction, and clustering.

Various data mining techniques have been applied for designing efficient intrusion detection systems because it has the advantage of discovering useful knowledge that describes a user's or program's behavior from large audit data sets. This papers proposes a Flexible Neural Tree (FNT) [1] for selecting the input variables and detection of network intrusions. Based on the predefined instruction/operator sets, a flexible neural tree model can be created and evolved. FNT allows input variables selection, over-layer connections and different activation functions for different nodes. In our previous work, the hierarchical structure was evolved using Probabilistic Incremental Program Evolution algorithm (PIPE) [2] with specific instructions. In this research work, the hierarchical structure is evolved using genetic programming. The fine tuning of the parameters encoded in the structure is accomplished using memetic algorithm.

The proposed method interleaves both optimizations. Starting with random structures and corresponding parameters, it first tries to improve the structure and then as soon as an improved structure is found, it fine tunes its parameters. It then goes back to improving the structure again and, fine tunes the structure and rules' parameters. This loop continues until a satisfactory solution is found or a time limit is reached.

2 A Memetic Algorithm

Memetic Algorithms are population-based approaches for heuristic search in optimization problems [3]. Basically, they are genetic algorithms that apply a separate local search process to refine individuals. One big difference between memes and genes is that memes are processed and possibly improved by the people that hold them - something that cannot happen to genes. Experimental results show that the memetic algorithms have better results over simple genetic or evolutionary algorithms [4].

2.1 Genetic Algorithm

Genetic algorithm based on the Darwinian survival of the fittest theory, is an efficient and broadly applicable global optimization algorithm [5]. In contrast to conventional search techniques, genetic algorithm starts from a group of points coded as finite length alphabet strings instead of one real parameter

set. Furthermore, genetic algorithm is not a hill-climbing algorithm hence the derivative information and step size calculation are not required. The three basic operators of genetic algorithms are: selection, crossover and mutation. It selects some individuals with stronger adaptability from population according to the fitness, and then decides the copy number of individual according to the selection methods such as Backers stochastic universal sampling. It exchanges and recombines a pair of chromosome through crossover. Mutation is done to change certain point state via probability. In general, one needs to choose suitable crossover and mutation probability time and again via real problems.

2.2 Local Search Method

Local search method is a method of searching a small area around a solution and adopting a better solution if found. The search begins with choosing a direction of movement is prescribed according to some algorithm, and a line search or trust region approach is performed to determine an appropriate next step. The process is repeated at the new point and the algorithm continues until a local minimum is found. In this research, a simple random local search model is employed.

Given a parameter vector $\theta(k) = [\lambda_1(k), \lambda_2(k), \ldots, \lambda_n(k)]$, where k is random search step. Let $x(k) = [x_1(k), x_2(k), \ldots, x_n(k)]$ denote the small random disturbance vector which is generated according to a probability density function. The random search algorithm used can be summarized as follows:

1) Choose an initial value of the parameter vector to be optimized randomly, $\theta(0)$, calculate the objective function, $F(\theta(0))$, and set $k = 0$.
2) Generate random search vector $x(k)$.
   - calculate $F(\theta(k) + x(k))$. If $F(\theta(k) + x(k)) < F(\theta(k))$, the current search is said to be success and $\theta(k+1) = \theta(k) + x(k)$. else,
   - calculate $F(\theta(k) - x(k))$. If $F(\theta(k) - x(k)) < F(\theta(k))$, the current search is said to be success too and $\theta(k+1) = \theta(k) - x(k)$. otherwise,
   - the search is said to be failure, and $\theta(k+1) =$

$$\begin{cases} \theta(k) \quad If\ K_{er}^{+} > K_{er}, K_{er}^{-} > K_{er} \\ \theta(k) + x(k)\ If\ K_{er}^{+} < K_{er}^{-} \\ \theta(k) - x(k)\ If\ K_{er}^{+} \geq K_{er}^{-} \end{cases} \tag{1}$$

where $K_{er} \geq 1$ is the maximum error ratio. The user defined parameter $K_{er} = 1.001$ for our experiments. K_{er}^{+} and K_{er}^{-} are defined by

$$K_{er}^{+} = \frac{F(\theta(k) + x(k))}{F(\theta(k))} \tag{2}$$

$$K_{er}^{-} = \frac{F(\theta(k) - x(k))}{F(\theta(k))} \tag{3}$$

3) If a satisfied solution is found then stop, else set $k = k + 1$ and go to step 2).

Note that the effectiveness of random search depends largely on the random search vector $x(k)$. Here, the $x(k)$ is created and adapted by using similar technique with the Evolution Strategy (ES).

$$\begin{aligned} x(k) &= N(0, \sigma(k)^2) \\ \sigma_i(k) &= \sigma_i(k-1) e^{\bar{\tau} z} e^{\tau z_i}, i \in 1, 2, \ldots, n \end{aligned} \tag{4}$$

where $x(k)$ is the parameter vector to be optimized and τ, $\bar{\tau}$ and σ_i are the strategy parameters. The values for τ and $\bar{\tau}$ are fixed to

$$\tau = (\sqrt{2\sqrt{n}})^{-1}, \bar{\tau} = (\sqrt{2n})^{-1} \tag{5}$$

The z_i and z are sampled from a standard normal distribution, which characterize the mutation exercised on the strategy and the objective parameters. The σ_i is also called step-sizes without self-adaption.

The proposed Memetic Algorithms can be described as follows:

S1 Generate an initial GA population;
S2 Evaluate all individuals in the population;
S3 For each individual in the population perform local search on it and Replace it with locally improved solution;
S4 Apply standard genetic algorithm operators to create a new population.
S5 If satisfactory solution is found then stop, otherwise go to step S2.

In this research, the memetic algorithm is employed to optimize the parameter vector of FNT and the weights and bias of a NN.

3 Flexible Neural Tree Classifier

In this research, a tree-structural based encoding method with specific instruction set is selected for representing a FNT model [1][6].

3.1 Flexible Neuron Instructor and FNT Model

The function set F and terminal instruction set T used for generating a FNT model are described as follows:

$$S = F \bigcup T = \{+_2, +_3, \ldots, +_N\} \bigcup \{x_1, \ldots, x_n\}, \tag{6}$$

where $+_i (i = 2, 3, \ldots, N)$ denote non-leaf nodes' instructions and taking i arguments. x_1,x_2,...,x_n are leaf nodes' instructions and taking no other arguments. The output of a non-leaf node is calculated as a flexible neuron model (see Figure 1). From this point of view, the instruction $+_i$ is also called a flexible neuron operator with i inputs. In the creation process of neural tree, if a nonterminal instruction, i.e., $+_i (i = 2, 3, 4, \ldots, N)$ is selected, i real values are randomly generated and used for representing the connection strength between the node $+_i$ and its children. In addition, two adjustable parameters a_i and b_i are randomly created as flexible activation function parameters. Some examples of flexible activation functions are shown in Table 1.

For developing the FNT classifier, the following flexible activation function is used.

$$f(a_i, b_i, x) = e^{-(\frac{x-a_i}{b_i})^2} \tag{7}$$

The output of a flexible neuron $+_n$ can be calculated as follows. The total excitation of $+_n$ is

$$net_n = \sum_{j=1}^{n} w_j * x_j \tag{8}$$

where $x_j (j = 1, 2, \ldots, n)$ are the inputs to node $+_n$. The output of the node $+_n$ is then calculated by

$$out_n = f(a_n, b_n, net_n) = e^{-(\frac{net_n - a_n}{b_n})^2}. \tag{9}$$

Table 1
The flexible activation functions

Gaussian Function	$f(x, a, b) = exp(-\frac{(x-a)^2}{b^2})$
Unipolar sigmoid function	$f(x, a) = \frac{2\|a\|}{1+e^{-2\|a\|x}}$
Bipolar sigmoid function	$f(x, a) = \frac{1-e^{-2xa}}{a(1+e^{-2xa})}$
Nonlocal radial coordinates	$f(x, a, b) = (b^2 + \|x - a\|^2)^{-\alpha} (\alpha > 0)$
General multiquadratics	$f(x, a, b) = (b^2 + \|x - a\|^2)^{\beta} (0 < \beta < 1)$
Thin-plate s-spline function	$f(x, a, b) = (b\|x - a\|)^2 \ln(b\|x - a\|)$

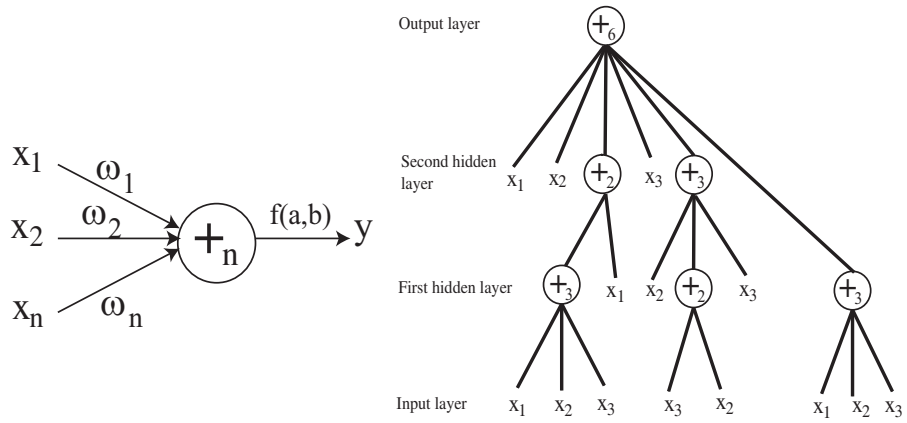

Fig. 1. A flexible neuron operator (left), and a typical representation of the FNT with function instruction set $F = \{+_2, +_3, +_4, +_5, +_6\}$, and terminal instruction set $T = \{x_1, x_2, x_3\}$ (right)

A typical flexible neuron operator and a neural tree model are illustrated in Figure 1. The overall output of flexible neural tree can be computed from left to right by depth-first method, recursively.

3.2 The Optimization of FNT Model

The optimization of FNT including the tree-structure and parameter optimization. Finding an optimal or near-optimal neural tree is formulated as a product of evolution. A number of neural tree variation operators are developed as follows:
Mutation: Five different mutation operators were employed to generate offspring from the parents.These mutation operators are as follows:

(1) Changing one terminal node: randomly select one terminal node in the neural tree and replace it with another terminal node;
(2) Changing all the terminal nodes: select each and every terminal node in the neural tree and replace it with another terminal node;
(3) Growing: select a random leaf in hidden layer of the neural tree and replace it with a newly generated subtree.
(4) Pruning: randomly select a function node in the neural tree and replace it with a terminal node.
(5) Pruning the redundant terminals: if a node has more than 2 terminals, the redundant terminals should be deleted.

Crossover: Select two neural trees randomly and select one nonterminal node in the hidden layer for each neural tree randomly, and then swap the selected subtree. The crossover operator is implemented with a pre-defined a probability 0.3 in this study.

Selection: Evolutionary programming (EP) style tournament selection was applied to select the parents for the next generation [7]. Pairwise comparison is conducted for the union of μ parents and μ offsprings. For each individual, q opponents are chosen uniformly at random from all the parents and offspring. For each comparison, if the individual's fitness is no smaller than the opponent's, it receives a selection. Select μ individuals out of parents and offsprings, that have most wins to form the next generation. This is repeated for each generation until a predefined number of generations or when the best structure is found.

3.2.1 Parameter Optimization by MA

Parameter optimization is achieved by the MA algorithm as described in Section 2. In this stage, the architecture of FNT model is fixed, and it is the best tree developed during the end of run of the structure search. The parameters (weights and flexible activation function parameters) encoded in the best tree formulate an individual. The GA algorithm works as follows:

(a) Initial population is generated randomly. The learning parameters crossover and mutation probabilities in MA should be assigned in advance.
(b) The objective function value is calculated for each individual.
(c) Implementation of the local search, selection, crossover and mutation operators.
(d) If maximum number of generations is reached or no better parameter vector is found for a significantly long time (100 steps), then stop, otherwise goto step (b).

3.3 Feature/Input Selection with FNT

It is often a difficult task to select variables (features) for the classification problem, especially when the feature space is large. A fully connected NN classifier usually cannot do this. In the perspective of FNT framework, the nature of model construction procedure allows the FNT to identify important input features in building an IDS that is computationally efficient and effective.

The mechanisms of input selection in the FNT constructing procedure are as follows. (1) Initially the input variables are selected to formulate the FNT model with same probabilities; (2) The variables which have more contribution to the objective function will be enhanced and have high opportunity to survive at next generation by an evolutionary procedure; (3) The evolutionary operators i.e., crossover and mutation, provide a input selection method by which the FNT should select appropriate variables automatically.

4 NN Classifier

A neural network classifier trained by MA with flexible bipolar sigmoid activation functions at hidden layer were constructed. Before describing details of the algorithm for training NN classifier, the issue of coding is presented. Coding concerns the way the weights and the flexible activation function parameters of NN are represented by individuals or particles. A float point coding scheme is adopted here. For NN coding, suppose there are M nodes in hidden layer and one node in output layer and n input variables, then the number of total weights is $n * M + M * 1$, the number of thresholds is $M + 1$ and the number of flexible activation function parameters is $M + 1$, therefore the total number of free parameters in a NN to be coded is $n * M + M + 2(M + 1)$. These parameters are coded into an individual or particle orderly.

The simple loop of the proposed training algorithm for neural network is as follows.

S1 Initialization. Initial population is generated randomly. The learning parameters, i.e., crossover and mutation probabilities, should be assigned in advance.
S2 Evaluation. The objective function value is calculated for each individual.
S3 Implementation of the local search, selection, crossover and mutation operators.
S4 if maximum number of generations is reached or no better parameter vector is found for a significantly long time (100 steps), then stop, otherwise goto step **S2**;

5 Decision Tree Classification

For comparison purpose, a Decision Tree (DT) classification method is also implemented. Feature selection is done based on the contribution the input variables make to the construction of the decision tree. Feature importance is determined by the role of each input variable either as a main splitter or as a surrogate. Surrogate splitters are defined as back-up rules that closely mimic the action of primary splitting rules. Suppose that, in a given model, the algorithm splits data according to variable protocol_type and if a value for protocol_type is not available, the algorithm might substitute service as a good surrogate. Variable importance, for a particular variable is the sum across all nodes in the tree of the improvement scores that the predictor has when it acts as a primary or surrogate (but not competitor) splitter. Example, for node i, if the predictor appears as the primary splitter then its contribution towards importance could be given as $i_{importance}$. But if the variable appears as the

Table 2
Parameter Settings

population size	50
crossover probability	0.4
mutation probability	0.03
α	0.7
$\beta(0)$	0.05
$\beta(1)$	0.4

n-th surrogate instead of the primary variable, then the importance becomes $i_{importance} = (p^n) * i_{improvement}$, in which p is the surrogate improvement weight which is a user controlled parameter set between (0-1)[12].

Decision tree induction is one of the classification algorithms in data mining. Classification algorithm is inductively learned to construct a model from the pre-classified data set. The inductively learned model of classification algorithm is used to develop IDS [22][12].

6 Experiment Results and Analysis

The parameters used for both experiments are listed in Table 2. In addition, all experiments were performed using an 2.8 GHz processor with 512 MB of RAM.

6.1 Intrusion Detection System

Intrusion detection is classified into two types: misuse intrusion detection and anomaly intrusion detection. Misuse intrusion detection uses well-defined patterns of the attack that exploit weaknesses in system and application software to identify the intrusions. Anomaly intrusion detection identifies deviations from the normal usage behavior patterns to identify the intrusion.

Various intelligent paradigms namely Neural Networks [8], Support Vector Machine [9], Neuro-Fuzzy systems [10], Linear genetic programming [11] and Decision Trees [12] have been used for intrusion detection. Various data mining techniques have been applied to intrusion detection because it has the advantage of discovering useful knowledge that describes a user's or program's behavior from large audit data sets.

Current Intrusion Detection Systems (IDS) examine all data features to detect intrusion or misuse patterns. Some of the features may be redundant or contribute little (if anything) to the detection process. The purpose of this

study is to identify important input features in building an IDS that is computationally efficient and effective. This paper proposes an IDS model based on general and enhanced Flexible Neural Tree (FNT).

6.1.1 Data set

The data for our experiments was prepared by the 1998 DARPA intrusion detection evaluation program by MIT Lincoln Lab. The data set contains 24 attack types that could be classified into four main categories namely *Denial of Service (DOS)*, *Remote to User (R2L)*, *User to Root (U2R)* and *Probing*. The original data contains 744 MB data with 4,940,000 records. The data set has 41 attributes for each connection connection record plus one class label. Some features are derived features, which are useful in distinguishing normal from attacks. These features are either nominal or numeric. Some features examine only the connection in the past two seconds that have the same destination host as the current connection, and calculate statistics related to protocol behavior, service, etc. These called same host features. Some features examine only the connections in the past two seconds that have same service as the current connection and called same service features. Some other connection records were also stored by destination host, and features were constructed using a window of 100 connections to the same host instead of a time window. These called host-based traffic features. R2L and U2R attacks don't have any sequential patterns like DOS and Probe because the former attacks have the attacks embedded in the data packets whereas the later attacks have many connections in a short amount of time. So some features that look for suspicious behavior in the data packets like number of failed logins are constructed and these are called contents features. The data for our experiments contains randomly generated 11982 records having 41 features [13].

This data set has five different classes namely *Normal DOS*, *R2L*, *U2R* and *Probe*. The training and test comprises of 5092 and 6890 records respectively. All the IDS models were trained and tested with the same set of data. As the data set has five different classes we performed a 5-class binary classification. The *normal* data belongs to class 1, *Probe* belongs to class 2, *DOS* belongs to class 3, *U2R* belongs to class 4 and *R2L* belongs to class 5.

6.1.2 Feature Selection and Classification Using FNT Paradigms

For this simulation, the original 41 input variables are used for constructing a FNT model. A FNT classifier was constructed using the training data and then the classifier was used on the test data set to classify the data as an attack or normal data. The instruction sets used to create an optimal FNT classifier is $S = F \bigcup T = \{+_5, \ldots, +_{20}\} \bigcup \{x_1, x_2, \ldots, x_{41}\}$. Where $x_i (i = 1, 2, \ldots, 41)$

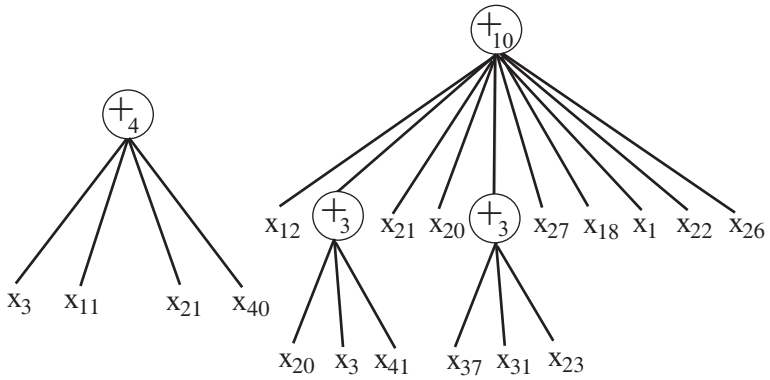

Fig. 2. The evolved FNT trees for classes 1 and 2 with 41 input variables

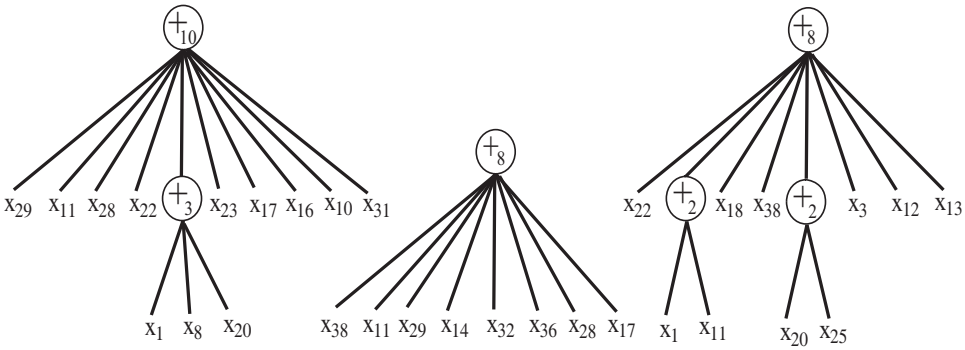

Fig. 3. The evolved FNT trees for classes 3, 4 and 5 with 41 input variables

denotes the 41 features.

The optimal FNTs for classes 1, 2, 3, 4, 5 are shown in Figures 2 and 3. It should be noted that the important features for constructing the FNT model were formulated in accordance with the procedure mentioned in the previous section. These important variables are shown in Table 3. Table 4 depicts the detection performance of the FNT by using the original 41 variable data set.

Table 3
The important features selected by the FNT algorithm

Attack Class	Important variables
Normal	x_3, x_{11}, x_{21}, x_{40}
Probe	x_1, x_3, x_{12}, x_{18}, x_{20}, x_{21}, x_{23}, x_{26}, x_{27}, x_{31}, x_{37}, x_{41}
DOS	x_1, x_8, x_{10}, x_{11}, x_{16}, x_{17}, x_{20}, x_{12}, x_{23}, x_{28}, x_{29}, x_{31}
U2R	x_{11}, x_{14}, x_{17}, x_{28}, x_{29}, x_{32}, x_{36}, x_{38}
R2L	x_1, x_3, x_{11}, x_{12}, x_{13}, x_{18}, x_{20}, x_{22}, x_{25}, x_{38}

Table 4
Detection performance using FNT, NN and DT classification models

Attack Class	FNT	NN	DT
Normal	**99.19%**	95.69%	82.32%
Probe	**98.39%**	95.53%	94.83%
DOS	**98.75%**	90.41%	77.10%
U2R	99.70%	**100%**	99.83%
R2L	**99.09%**	98.10%	94.33%

Table 5
Comparison of false positive rate (fp) and true positive rate (tp) for FNT, NN and DT classifiers

Cancer	FNT		NN		DT	
Type	fp(%)	tp(%)	fp(%)	tp(%)	fp(%)	tp(%)
Normal	0.98	65.00	4.3	88.70	29.66	99.60
Probe	0.18	55.86	0.40	37.15	0.24	31.00
DOS	1.59	98.98	3.68	89.38	72.10	97.63
U2R	0.16	32.00	0.05	55.81	0.022	59.26
R2L	0.17	90.76	0.17	86.63	0.022	30.73

6.1.3 Classification Using NN Without Feature Selection

For comparison purpose, a neural network classifier trained by MA with flexible bipolar sigmoid activation functions were constructed using the same training data sets and then the neural network classifier was used on the test data set to detect the different types of attacks. All the input variables were used for the experiments. Table 4 depicts the detection performance of NN by using the original 41 variable data set.

6.1.4 Decision Tree Classification

The important variables were also decided by their contribution to the construction of the decision tree. Variable rankings were generated in terms of percentages. We eliminated the variables that had 0.00% rankings and considered only the primary splitters or surrogates. This resulted in a reduced 12 variable data set with x_2, x_4, x_5, x_{11}, x_{22}, x_{23}, x_{24}, x_{27}, x_{30}, x_{31}, x_{32}, x_{34} as variables. The detection performance of the DT by using the original 41 variable data set is shown in Table 4.

Receiver Operating Characteristics (ROC) analysis of the FNT, NN and DT models are shown in Table 5.

6.2 Breast Cancer Classification

Breast cancer is the most common cancer in women in many countries. Most breast cancers are detected as a lump/mass on the breast, or through self-examination or mammography [14]. Screening mammography is the best tool available for detecting cancerous lesions before clinical symptoms appear [20]. Surgery through a biopsy or lumpectomy have been also been the most common methods of removal. Fine needle aspiration (FNA) of breast masses is a cost-effective, non-traumatic, and mostly invasive diagnostic test that obtains information needed to evaluate malignancy. Recently, a new less invasive technique, which uses super-cooled nitrogen to freeze and shrink a non-cancerous tumor and destroy the blood vessels feeding the growth of the tumour, has been developed [15] in the USA.

Various artificial intelligence techniques have been used to improve the diagnoses procedures and to aid the physician's efforts [16][17][18][19].

6.2.1 Data sets

As a preliminary study, we made use of the Wisconsin breast cancer data set from the UCI machine-learning database repository [21]. This data set has 32 attributes (30 real valued input features) and 569 instances of which 357 are of benign and 212 are of malignant type. We randomly divided the training and test data sets. The first 285 data is used for training and the remaining 284 data is used for testing the performance of the different models.

6.2.2 Results

All the models were trained and tested with the same set of data. As the data set has two different classes, we performed a 2-class binary classification. The instruction sets used to create an optimal FNT classifier is $S = F \bigcup T = \{+_5, \ldots, +_{15}\} \bigcup \{x_0, x_1, \ldots, x_{29}\}$. Where $x_i (i = 0, 1, \ldots, 29)$ denotes the 30 input features. The optimal FNTs for classes 1 and 2 are shown in Figure 4. Table 7 depicts the classification performance of the FNT, NN and DT by using the original 30 input variables data set. It should be noted that the obtained FNT classifier has smaller size and reduced features and without a significant reduction in the accuracy. The important features for constructing the FNT models are shown in Table 6. The important variables by DT are x_6, x_7, x_{18}, x_{20}, x_{22}, x_{26}, x_{27}, x_{28} and x_{29}. Receiver Operating Characteristics (ROC) analysis of the FNT, NN and DT models are shown in Table 8.

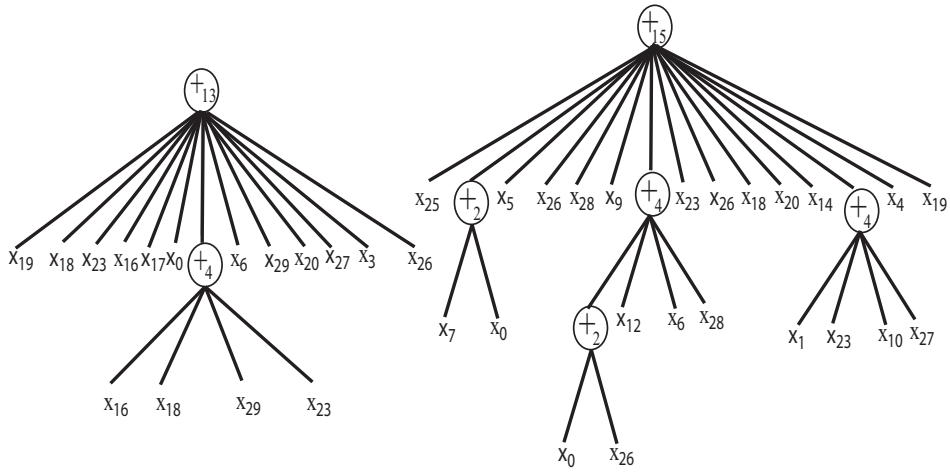

Fig. 4. The evolved FNT trees for Benign (left) and Malignant (right) classifications with 30 input variables

Table 6
The important features selected by the FNT algorithm

Cancer type	Important variables
Benign	x_0,x_3,x_6,x_{16},x_{17},x_{18},x_{19},x_{20},x_{23},x_{26},x_{27},x_{29}
Malignant	x_0,x_1,x_4,x_5,x_6,x_7,x_9,x_{10},x_{12},x_{14},x_{18},x_{19}, x_{20},x_{23},x_{25},x_{26},x_{27},x_{28}

Table 7
Comparative results of the FNT, NN and DT classification methods for the detection of breast cancer

Cancer type	FNT(%)	NN(%)	DT(%)
Benign	**94.72**	94.72	93.66
Malignant	**94.75**	94.01	93.66

Table 8
Comparison of false positive rate (fp) and true positive rate (tp) for FNT, NN and DT classifiers

Cancer	FNT		NN		DT	
Type	fp(%)	tp(%)	fp(%)	tp(%)	fp(%)	tp(%)
Benign	4.59	97.16	6.80	95.58	11.93	97.15
Malignant	1.70	83.49	3.87	90.29	2.84	88.07

7 Conclusion

In this paper, we presented a Flexible Neural Tree (FNT) model for Intrusion Detection Systems (IDS) and breast cancer classification with a focus on improving the detection/classification performance by reducing the input features.

As evident from Tables 4 and 7, the proposed flexible neural tree approach

seems to be very promising. The FNT model was able to reduce the number of variables to 4, 12, 12, 8 and 10 (using 41 input variables) for classes 1-5 respectively. Using 41 variables, FNT model gave the best accuracy for the DOS and U2R. Using 41 input variables, the direct NN classifier outperformed the FNT approach for U2R attack only, and the DT classifiers gave the best accuracy for Normal and Prob classes which are slightly better than the FNT classifiers.

Similarly for breast cancer classification problem, the FNT model was able to reduce the number of variables to 12 and 18 (using 30 input variables) without significant reduction in the classification accuracy.

Even though our current research was focused only on IDS and breast cancer classification, we believe that the proposed model could be extensively used for other classification problems involving numerous input features. We are also planning to investigate this in our future research.

Acknowledgments

This research was partially supported by the Natural Science Foundation of China under grant number 60573065, and The Provincial Science and Technology Development Program of Shandong under grant number SDSP2004-0720-03.

Authors would like to thank the anonymous referees for the technical suggestions and remarks which helped to improve the contents and the quality of presentation.

References

[1] Chen, Y., Yang, B., Dong, J., Nonlinear systems modelling via optimal design of neural trees. International Journal of Neural ystems. **14**, (2004) 125-138

[2] Salustowicz, R. P. and J. Schmidhuber, Probabilistic Incremental Program Evolution, *Evolutionary Computation*, Vol.2, No.5, pp. 123-141, 1997.

[3] Moscato, P., On evolution, search, optimization genetic algorithms and martial arts: towards memetic algorithms, Caltech Concurrent Computation Program, C3P Report 826, 1989.

[4] Hart, W.E., Adaptive global optimization with local search, Phd thesis, university of Califonia, San Diego, May 1994.

[5] Goldberg, D.E., Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley Publishing Company, Inc, 1989.

[6] Chen, Y., Yang, B., Dong, J., Abraham A.: Time Series Forecasting Using Flexible Neural Tree Model, Information Sciences, Elsevier Science, Vol. 174, Issues 3/4, pp. 219-235, 2005.

[7] Chellapilla, K., Evolving computer programs without subtree crossover, IEEE Transactions on Evolutionary Computation, Vol.1, 209-216, 1997.

[8] Debar M., Becke D., and Siboni A.: A Neural Network Component for an Intrusion Detection System. Proceedings of the IEEE Computer Society Symposium on Research in Security and Privacy, 1992.

[9] Mukkamala S., Sung A.H. and Abraham A.: Intrusion detection using an ensemble of intelligent paradigms, Journal of Network and Computer Applications, Elsevier Science, Vol. 28, Issue 2, (2005) 167-182.

[10] Shah K., Dave N., Chavan S., Mukherjee S., Abraham A. and Sanyal S.: Adaptive Neuro-Fuzzy Intrusion Detection System. IEEE International Conference on ITCC'04, Vol. 1, (2004)70-74.

[11] Abraham A.: Evolutionary Computation in Intelligent Web Management. Evolutionary Computing in Data Mining, A. Ghosh and L. Jain (Eds.), Studies in Fuzziness and Soft Computing, Springer Verlag Germany, Chapter 8, (2004)189-210.

[12] Chebrolu S., Abraham A., Thomas J.P.: Feature Deduction and Ensemble Design of Intrusion Detection Systems, Computers and Security, Elsevier Science, Volume 24/4, pp. 295-307, 2005.

[13] KDD cup 99 data set, http://kdd.ics.uci.edu/database/kddcup99/.

[14] DeSilva, C.J.S. et al., Artificial Neural networks and Breast Cancer Prognosis, The Australian Computer Journal, 26, pp. 78-81, 1994.

[15] The Weekend Australia, Health Section, pp. 7. July, 13-14, 2002.

[16] David B. Fogel , Eugene C. Wasson , Edward M. Boughton and Vincent W. Porto, A step toward computer-assisted mammography using evolutionary programming and neural networks, Cancer Letters, Volume 119, Issue 1, pp. 93-97, 1997.

[17] Charles E. Kahn, Jr , Linda M. Roberts, Katherine A. Shaffer and Peter Haddawy, Construction of a Bayesian network for mammographic diagnosis of breast cancer, Computers in Biology and Medicine, Volume 27, Issue 1, pp. 19-29, 1997.

[18] Shinsuke Morio , Satoru Kawahara , Naoyuki Okamoto, Tadao Suzuki, Takashi Okamoto , Masatoshi Haradas and Akio Shimizu, An expert system for early detection of cancer of the breast, Computers in Biology and Medicine, Volume 19, Issue 5, pp. 295-305, 1989.

[19] Barbara S. Hulka and Patricia G. Moorman, Breast Cancer: Hormones and Other Risk Factors, Maturitas, Volume 38, Issue 1, pp. 103-113, 2001.

[20] Jain, R. and Abraham, A., A Comparative Study of Fuzzy Classifiers on Breast Cancer Data, Australiasian Physical And Engineering Sciences in Medicine, Australia, Volume 27, No.4, pp. 147-152, 2004.

[21] Merz J., and Murphy, P.M., UCI repository of machine learning databases, http://www.ics.uci.edu/- learn/MLRepository.html, 1996.

[22] Brieman L., Friedman J., Olshen R., and Stone C., Classification of Regression Trees. Wadsworth Inc., 1984.