ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645
Indian Journal of Science and Technology, *Vol 10(18), DOI: 10.17485/ijst/2017/v10i18/92674, May 2017*

Improved Learning in Constructive Neural Networks for Pattern Classification using Genetic Algorithm

Mochitha Vijayan*, S. S Sridhar and Vivia John

Department of Computer Science and Engineering, SRM University, Chennai – 603203, Tamil Nadu, India; 02mochitha@gmail.com

Abstract

Objective: To remove rogue entries in the data set chose for training thus making the Neural Network (NN) learning easier. **Methods/Analysis:** In this research work Genetic Algorithm (GA) is being used on the data sets Wine, Pima Indians Diabetes, Iris, Vehicle and Image Segmentation from the UC Irvine machine learning repository. The refined data sets are then fed to a Constructive Neural Network (CoNN) for pattern classification task. CoNN dispenses an optimal strategy in designing the structures of the multilayer perceptron networks, coupled with supervised algorithms for further learning. They eliminate the requisite of ad-hoc, apriori design of network architecture which is often inappropriate. The M-Tiling algorithm is used for constructing the network and a simple perceptron learning rule for training individual TLUs (Threshold Logic Unit). The CoNNs potential capability is in constructing networks with a network size adequate to the complexity level of the task and in attaining a satisfactory level of efficiency. **Findings**: Acquiring data from real world problems may result in the frequent occurrence of rogue values. Rogue values result in poor data quality and mislead the learning curve yielding imprecise NN models. A template string prototype is generated with the continual application of GA on the input data. This template string prototype removes the rogue values from the actual training set which results in the reduction of the size of input data set, in turn, makes learning easier. CoNN coupled with GA classifies patterns in Wine, Pima Indians Diabetes, Iris, Vehicle and Image Segmentation datasets available in UC Irvine machine learning repository with faster convergence, more generalization accuracy, and less space. The proposed work is unique since GA is exerted to the training data set before it is fed to the CoNN for learning. The proposed work also demonstrates the improvement of CoNN in terms of convergence, generalization accuracy, and space optimization. In this way, these experimental results add value to the currently existing work. **Improvements**: The proposed model can be further revamped by using other CoNN learning strategies.

Keywords: Artificial Neural Network, Constructive Neural Network, Genetic Algorithm, Pattern Classification

1. Introduction

Neural Networks, more specifically Artificial Neural Networks (ANNs) are robust models for machine learning with an ability to generalize, adapt and perform parallel computation which is the very picture of a human brain. This model consists of basic computational components called "neurons" fitted in a topology, sending signals (data) to others over a number of weighted connections. ANNs can be made adaptive and capable of learning by fine tuning these numeric weights with respect to the experience. The adaptive nature of these networks where "learning from experience" replaces "programming" in solving

**Author for correspondence*

problems makes it significant in the machine learning community. ANN algorithms help to train these numeral weights using a methodical stratagem. “Learning” with ANN allude to the search for an optimized topology and weights for the neural net to accomplish a specific task. A “layer” in ANN has a common job. The neuron count in the input, output and hidden layers and its service is problem dependent and the learning algorithm used. The input layer neuron count is generally equal to the attribute count in the training instances. The output layer neuron count is equal to the number of categories looked for. Resoluteness of the hidden layer neuron count lies on the intrinsic complexity of the problem and the training instance count.

Pattern classification is a real to M-ary mapping where y (function’s output) has to be one of the M ($M \geq 2$) categories (or classes) to be sought. It is a case of task-specific function approximation. An NN for predicting the class membership of data instances typically has N neurons in the input layer and M neurons in the output layer. The output neuron k ($1 \leq k \leq M$) acquires an ability to output ‘1’ during the training (whereas, other output neurons are trained to output ‘0’) for data instances that fall in the kth class. Individual neurons can be modelled by a simple mathematical (threshold) or hard-limiting transfer (activation) function.

Researchers have proposed a wide range of ANN architectures and algorithms, among all the Constructive Neural Network (CoNN) suggest a winning frame for multi-class pattern recognition problems. The benefit of CoNN algorithms over their counterparts is needless to assume a network topology prior to the training. The simultaneous occurrence in the dynamic creation of network’s hidden layer(s) and the training makes it distinctive. Starting from a small network these algorithms allows the network growth dynamically by connecting and training TLU’s (neurons) as required until a satisfactory convergence is achieved.[1] That is, a CoNN algorithm is contingent upon the TLU training algorithm deployed. For training a TLU, a constructive algorithm usually deploys the perceptron learning rule or any of its derived forms.[2] Training involves determining appropriate weights for the interconnections. Conventional neural network learning involves updating the weights on a network modelled prior to the training. Determining the network architecture in such a case is a challenge, which necessitates expensive guess and check. The approach of fixed network topology is useful only if the architecture is correctly chosen for the problem on hand. However, lack of efficient methodologies for determining dynamic optimal network topology makes this process difficult. Too small networks will be inadequate for the ANN’s to learn with expected accuracy, while a large network which tends to over fit the data instances will generally have poor performance on new data.

Instead of learning on a pre-fixed network topology, an algorithm which is constructive in nature learns the topology in a problem dependent way. This learning process automatically fits the topology size to the input avoiding overspecialization, but results in desirable generalization in pattern classification and hence has faster convergence in learning.[3]

Constructive Neural Networks facilitates an optimal way to build optimized networks. Simple Threshold Logic Units (TLU) forms the computing units. Starting with a single TLU, it adds TLUs whenever needed. CoNN is popular in two subtasks: Firstly, when current topology fails to achieve the result, it incrementally adds threshold neurons to the network resulting in the dynamic growth of the network. Secondly, it uses varied types of training algorithms to train the newly added threshold neuron(s).

The removal of rogue values is done by the template strings produced by Genetic Algorithm (GA). In the domain of artificial intelligence, GA is presented as a heuristic search strategy which is related to the natural selection process that mimics the biological evolution based on Darwinian survival of the fittest.[4-7] The iterative use of this heuristics generates useful solutions to search and optimization problems. Genetic algorithms are a special class of Evolutionary Algorithms (EA), which generate solutions using techniques such as inheritance, crossover, mutation, and selection.

Template strings generated by GA will set a dataset prototype. These Template strings are matched with the actual data set and refine it by removing the rogue values, which results in the reduction of the size of input data set, in turn, make learning easier. CoNN coupled with GA classifies the patterns in Wine dataset, Pima Indians Diabetes dataset, Iris dataset, Vehicle dataset and Image Segmentation dataset available in UCI machine learning repository with faster convergence, more generalization performance and with less spatial requirements.

Everett Fall, et al, presented a new approach in modelling neural networks.[8] The structure of proposed NN is not stringently defined (each neuron may receive input from any other neuron). The initial network structure is being generated at random, and traditional methods of training, such as back propagation, are replaced or augmented by a Genetic Algorithm (GA). The weight of all the input neurons is encoded to serve as the genes for the GA. By means of the training data provided to the supervised topology, the contribution of each neuron in creating the desired output serves as a selection strategy. Modifications are made on neurons to store, recall the old weighting which can be used in future. The proposed NN can mimic the flexibility of biological networks in adapting to unexpected data. A simple network is trained to identify vertical and horizontal lines as a proof of concept.

S. S. Sridhar, et al, presents several CoNN learning algorithms that facilitate the NN to be built along with the learning process.[9] The paper examines several algorithms like MTower, MPyramidal, MTower, MTiling, MSequential, MUpstart, Dist AI, MPerceptroncascade, and algorithms like correction procedure and the Pocket algorithm is used for training individual TLU's which have been proved to converge on Multi-categories. Constructive learning aims to automate the construction of a network topology of appropriate size provided the full training data set. Constructive algorithms feature a striking method to the self-acting design of neural networks for pattern classification. It avoids the need of choosing the architecture beforehand and provides a way for constructing networks with a problem dependent size. This paper is focussing on a set of algorithms that is cumulatively constructing networks of TLU's that which deals with multi categories.

Rajesh Parekh, et al, has worked on a group of algorithms that cumulatively construct feed forward networks of TLU's.[10] The proposed algorithms are limited to two-class pattern classification tasks with binary/bipolar input attributes. Two constructive learning algorithms MTiling-real and MPyramid-real are being presented. These algorithms extend the pyramid and the tiling algorithms, respectively, to handle multi- class classification of patterns that have real-valued attributes. For each of these algorithms, they have provided meticulous proofs of convergence to zero classification errors on finite, no contradictory training sets. This proof is general in nature. Additionally, they have shown how the local pruning step can remove the neuron redundancy from MTiling-real networks.

Rajesh Parekh, et al, provides convergence proofs for several constructive learning algorithms.[1] These algorithms offer a winning approach for the cumulative building of efficient minimal neural network architecture for pattern classification tasks. These algorithms overcome the necessity for an ad-hoc and often improper choice of a network topology for the algorithms that in search for appropriate weight in a prior fixed network architecture. The convergence proof for the Tower, Pyramidal, Tiling, and Sequential Learning (except Upstart, Perceptron cascade) relies on the assumption that the pattern attributes values are either binary or bipolar. This paper focuses on multi-categorical extensions of CoNN learning algorithms for the task of classification in which the input may contain real-valued attributes.

S. S. Sridhar, et al, proposes a new Tiling architecture for Constructive Neural Networks.[11] In the study, they introduced a new MTiling topology with unsupervised learning stratagems on two-class datasets for attaining efficient performance in respect of generalization, converging speed, and fewer weight connections and storage requirements.

S. S Sridhar, *et al.,* proposed an Improved learning which is adaptive in nature for Multi-class Tiling Constructive Neural Networks for classification problems.[12] The performance of various algorithms such as Tower, Pyramid, Tiling and New Tiling networks were surveyed using the datasets in UCI repository like Iris, B-Pattern dataset, Pima Indian Diabetic. Results show that the Multi Category Tiling network topology with revamped learning algorithm excels other network topologies. The study was on the basis of parameters such as the hidden layer and the neuron count, the number of patterns being classified for the generalization and attuning of misclassified patterns for better convergence. As per the outcomes, the revamped adaptive learning algorithm could be useful in various pattern classification tasks which involve constructive neural networks.

Philipp Koehn brings the idea of the genetic algorithm applied to neural networks. The combination has given good results for weight as well as architecture optimization.[13] As genetic algorithms work on the fitness function, they can be exerted to the large class of problems. For a large class of similar problems, the application of GANN architecture optimization is feasible if the goal is a good network topology. The high computation time is justified, as it has to be performed only once. Then, for a specific instance of the problem, the optimized topology can be used.

2. Materials and Methods

Constructive neural networks use the multi-categorical extensions of CoNN learning algorithms for the task of classification. Data acquired from the real world problems may have a frequent occurrence of rogue values. Rogue values result in poor data quality and mislead the learning curve yielding imprecise NN models. i.e. the qualitative aspect of the data may be affected by errors or variance caused during the data collection phase, as an out-turn of human error while information translation or because of the measurement equipment tolerance limitations. This may cause disparity in the attribute values or in the instances.

The rogue values can bias the learning process that which hinders the path of learning algorithms to build accurate NN models for the data provided. This work addresses this issue.

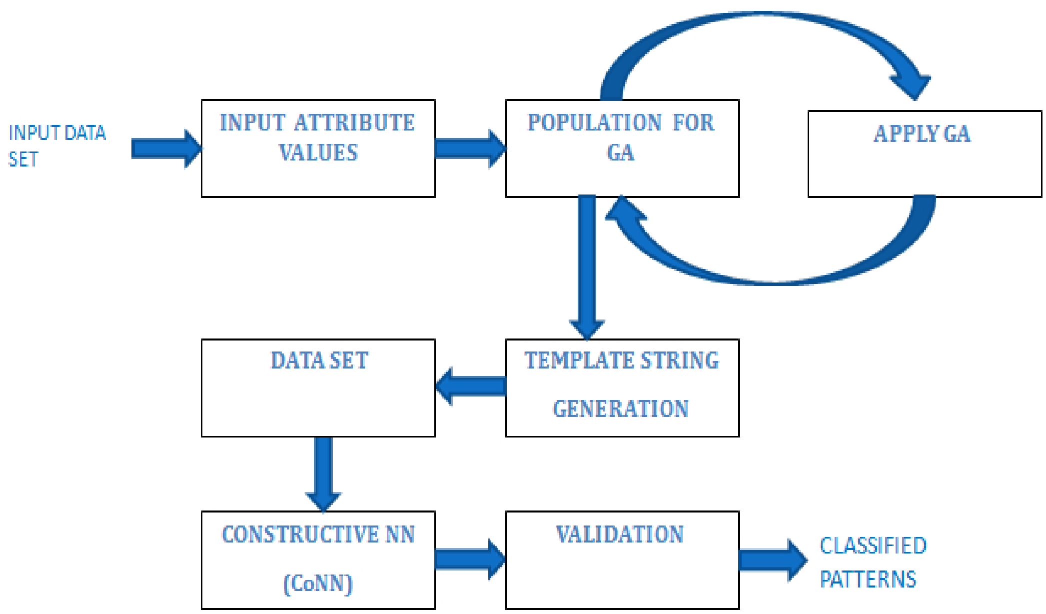

Figure 1. Block diagram of proposed system.

Proposed work presents a CoNN which dispenses an optimal strategy in designing the topology a multi-layer neural network, coupled with supervised learning algorithms for pattern classification tasks. Instead of proffering the network with input dataset that contains rogue values, we remove the rogue values prior to the training process using GA. This can lead the NN learning process in a right direction resulting in an accurate CoNN model which has faster convergence to the solution, more predictive performance and that which requires less space. Figure 1 is the block diagram of the proposed system.

2.1 Template String Generation using GA

A Genetic Algorithm (or GA) is a search technique used in computing to find true or approximate solutions to optimization and search problems.[14] GA comes under the class of evolutionary algorithms that uses nature-inspired techniques (evolutionary biology) such as inheritance, mutation, selection, and crossover (also named as recombination).[15]

The evolution is being initiated from a population of individuals which are generated arbitrarily and occurs in generations. Here the inputs from the dataset are considered as initial population.

“Fitness” of every individual (candidate) is evaluated in each of the, based on the fitness factor multiple individuals are selected from the current candidate population and modified to generate a new population. The contribution of each attribute of a data (individual) in making the data lie in an acceptable range act as a fitness strategy. Scores are assigned to each of the candidates according to their fitness. Individuals are sorted in the order of their scores. Candidates with top scores are eligible for selection and these individuals are considered as fit for the next generation. This newly generated population is used for the next iteration of algorithm. The algorithm aborts when the required number of template strings is achieved. Figure 2 demonstrates the GA process.

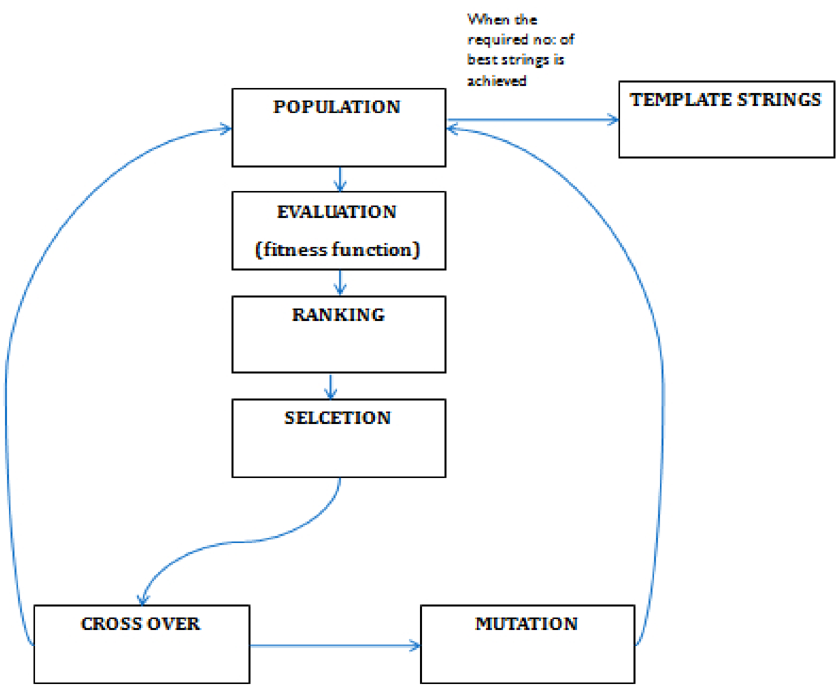

Figure 2. Genetic Algorithm for Template string Generation.

2.2 Refining the Actual Dataset using Template String

Template strings are considered as the benchmark datasets. The actual dataset is refined using the template string generated using GA. Only those datasets that match with the template strings is being refined. This process of refinement will remove the rogue values thereby improving the quality of data fed for training.

2.3 Constructive Neural Network Learning Algorithm

In order to classify a set of input patterns into one of two classes, a single perceptron or TLU shall be trained. This perceptron computes a function which is the weighted sum of inputs. The Multi-category Tiling algorithm incrementally constructs networks of threshold neurons to handle multi categories as discussed here. This algo-

Algorithm MTiling

a. Initialize a layer with O master neurons each of which is connected to N input neurons and train them.
b. If the desired classification is achieved by master neurons, then stop.
c. else, if the layer is not satisfiable,
Then, add ancillary neurons to this layer to make it satisfiable as follows,
Else go to step d.
1. Identify the output vectors for which maximum number input map to (an output vector is not satisfiable if it is generated by input belonging to different classes)
2. Identify the patterns that generate the output vector identified in step c (1). This patterns will be considered as the training set for ancillary neurons.
3. Add and train a set of n ($1 \leq n \leq O$) ancillary neurons, where n is the target class count represented in the pattern set identified above.
4. Repeat the last three steps until the output layer representation of the patterns is satisfiable.
d. The new layer of master neurons are trained and is connected to each neuron in the preceding layer, then go to step b.

Pseudo-code 1. M-Tiling algorithm.

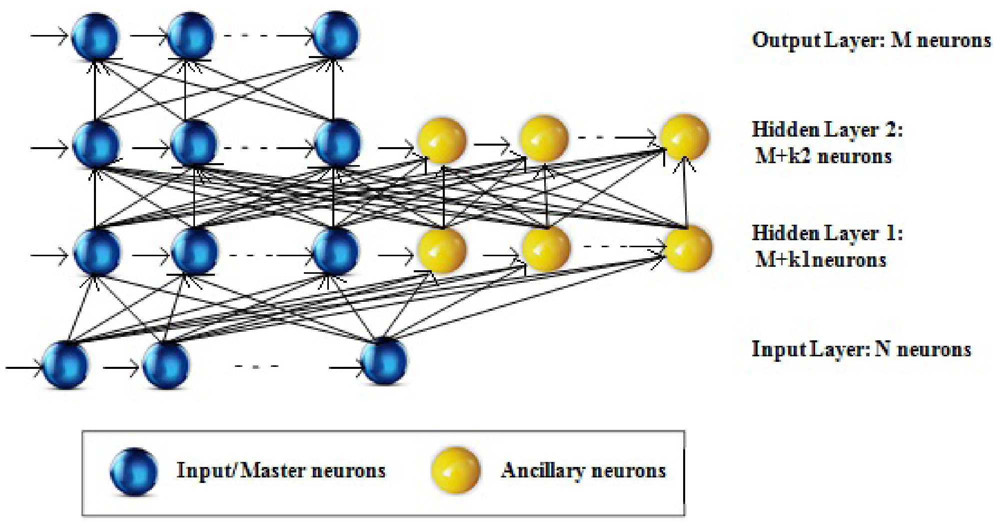

Figure 3. A CoNN using MTiling algorithm.

rithm builds a layered network of TLUs (Threshold Logic Units). The bottom-most layer receives inputs from each of the N input neurons. The neurons in each layer accept inputs from the layer of neurons immediate below to it; in this process, each layer holds a master neuron (output of that layer). In addition to master neurons, ancillary neurons are added to the layers for faithful representation, if classification is not done properly. Figure 3 is a CoNN constructed using MTiling algorithm.[11] MTiling algorithm is shown in pseudo-code 1.

A simple perceptron learning rule is used for training individual TLU's.

2.4 Validation

Validation phase involves the performance test on constructed CoNN by feeding new data into it for classification. If the training is done in the right way, it should work for the new cases as well as the one's we trained it on. Otherwise the training has to be done again until the required accuracy is being achieved and the patterns are classified correctly.

3. Results and Discussion

The performance of the existing CoNN and the new CoNN with GA is examined in this section. The real-world datasets Wine, Pima Indian diabetic, Iris, Vehicle and Image Segmentation are available at the UCI Machine Learning Repository.[16]

Different criteria for analysis are:

1. Misclassification count
2. Count of Hidden Layers
3. Count of Hidden Layer Neurons
4. Count of Ancillary Neurons
5. Total number of Neurons
6. Total Weight Connections
7. Time in ms

All the datasets are initially normalized to distribute the data evenly and scale it into an acceptable range for the network.[17] GA applied to the datasets has removed the rogue values thereby reducing the instance count in actual dataset. The number of instances in each dataset before

Table 1. Result of GA applied on dataset

Dataset	No: Of Training Dataset Instances	No: Of Refined Instances	No: Of Removed Instances
Wine	178	144	34
Pima	768	736	32
Iris	150	109	41
Vehicle	846	661	185
Image	2310	2286	24

and after applying GA, number of removed instances on using GA is being tabulated in Table 1.

The normalized attribute values of each training data set are fed to the CoNN to make it learn. Equal number of datasets is used for training the existing CoNN and the new CoNN with GA, and equal number of datasets from each specific category class is taken for evaluation in both the cases. Parameters like the misclassification count, hidden layer count, count of neurons in each hidden layer, ancillary neuron count, total number of neurons,

Table 2. Performance on Wine dataset

	CoNN	CoNN(GA)
Misclassifications	3	0
Hidden Layers	3	2
Hidden Layer Neurons	15	9
Ancillary Neurons	6	3
Total Neurons	31	25
Total Weight Connections	130	97
Time in ms	15.3	12.57

Table 3. Performance on Pima Indian Diabetes dataset

	CoNN	CoNN(GA)
Misclassifications	**1**	**0**
Hidden Layers	2	1
Hidden Layer Neurons	7	3
Ancillary Neurons	3	1
Total Neurons	17	13
Total Weight Connections	50	30
Time in ms	9.56	9.00

Table 4. Performance on Iris dataset

	CoNN	CoNN (GA)
Misclassifications	3	0
Hidden Layers	2	1
Hidden Layer Neurons	10	4
Ancillary Neurons	4	1
Total Neurons	17	11
Total Weight Connections	60	28
Time in ms	10.68	10

Table 5. Performance on Vehicle dataset

	CoNN	CoNN(GA)
Misclassifications	8	1
Hidden Layers	4	4
Hidden Layer Neurons	32	23
Ancillary Neurons	16	7
Total Neurons	54	45
Total Weight Connections	368	243
Time in ms	181.61	118.2

Table 6. Performance on Image Segmentation dataset

	CoNN	CoNN(GA)
Misclassifications	5	1
Hidden Layers	3	3
Hidden Layer Neurons	30	26
Ancillary Neurons	9	5
Total Neurons	56	52
Total Weight Connections	460	390
Time in ms	17.29	16.84

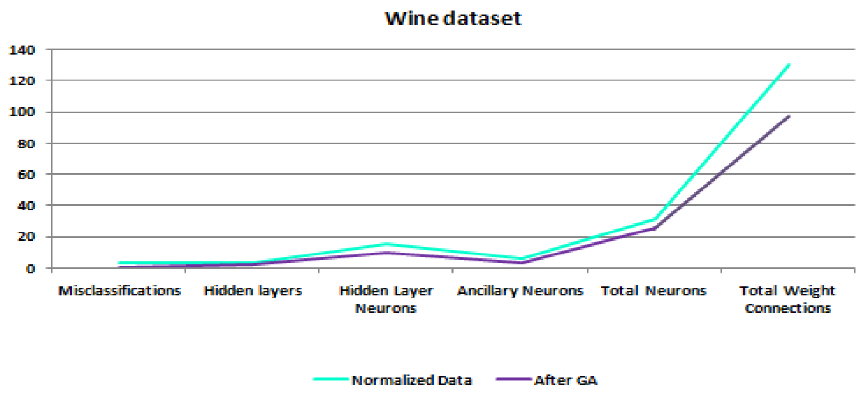

Figure 4. Performance on Wine dataset.

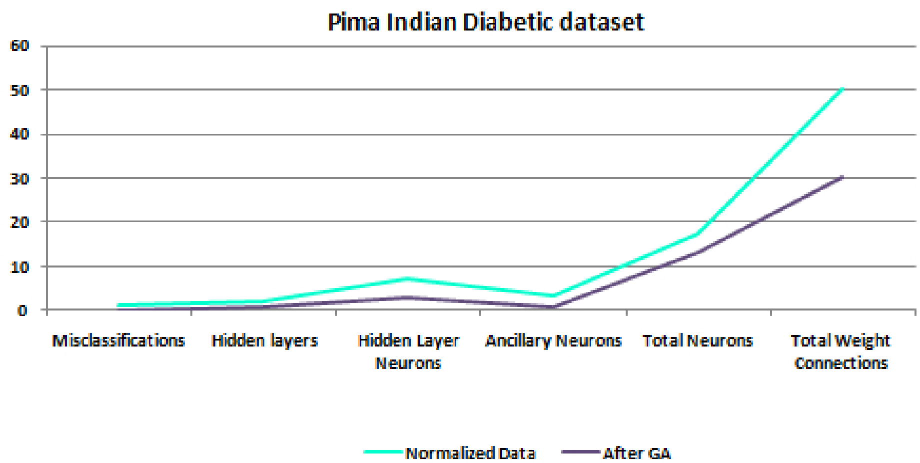

Figure 5. Performance on Pima Indian Diabetes dataset.

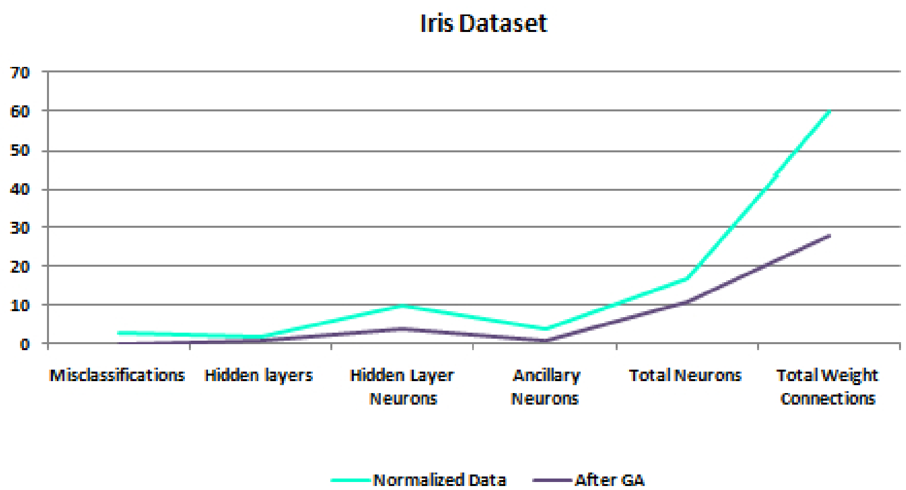

Figure 6. Performance on Iris dataset.

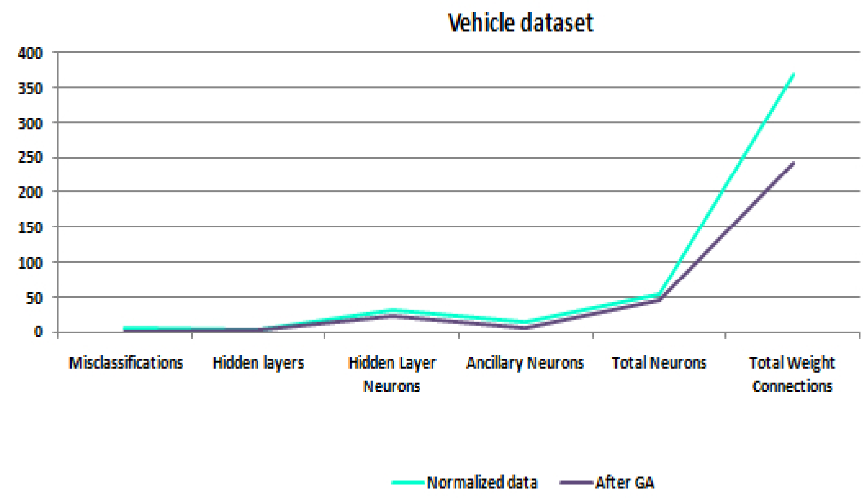

Figure 7. Performance on Vehicle dataset.

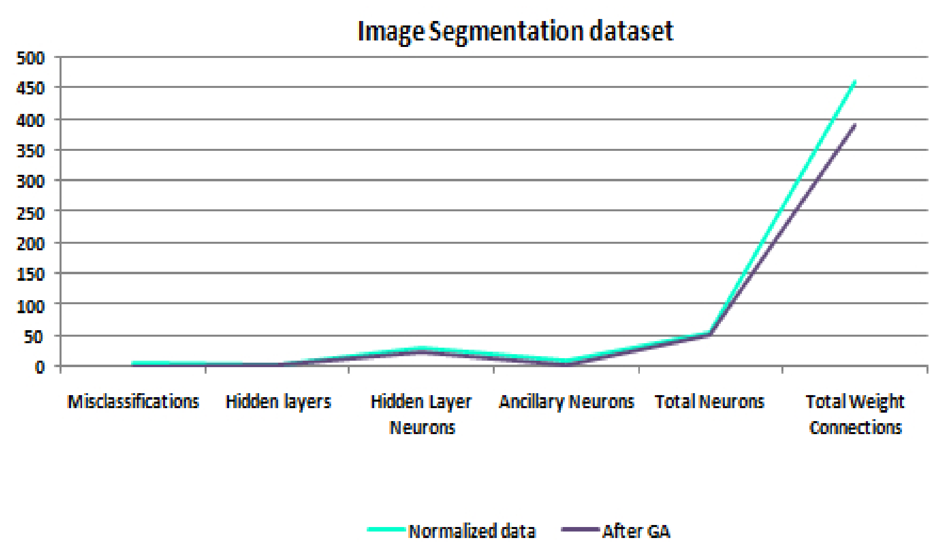

Figure 8. Performance on Image Segmentation dataset.

total weight connections and time taken for learning are being analyzed with respect to the outcomes of network evaluation. Table 2, 3, 4, 5 and 6 précis the outcomes of experiments done to examine the performance of CoNN and CoNN with GA on various datasets.

It is being analyzed that, in terms of all the criteria's (listed above) CoNN with GA succeeds the existing CoNN for all the datasets experimented. The performance graphs of Wine, Pima Indian Diabetes, Iris, Vehicle and Image Segmentation datasets are shown in Figure 4, 5, 6, 7 and 8 respectively.

4. Conclusion

In this work, we reduced the generalization errors, learning time and spatial requirements of a CoNN by applying GA to the input data set prior to the training process. The template string generated by GA removed rogue values from the actual training set reducing the size of the input data set and thereby making learning easier. This leads the constructive training algorithms to generate an accurate CoNN model which has faster convergence to the solution with less space and more generalization accuracy.

Following are some of the future research directions

1. The performance of CoNN with GA with respect to the criteria's such as the network size, training time, generalization capability and convergence properties on various other datasets of UCI repository can be done.
2. Other learning algorithms shall be experimented for improving the performance of CoNN with GA.
3. Application of diverse pre-processing techniques to the input dataset for improved performance.

5. Acknowledgments

This research was supported by SRM University, Kattankulathur, Chennai, Tamil Nadu. We are thankful to our college for providing expertise that greatly assisted the research. We would also like to show our gratitude to the Editorial committee members for reviewing this manuscript.

6. References

1. Parekh R, Yang J, Honavar V. Constructive neural network learning algorithm for multi-category real valued pattern classification. Computer Science Technical Reports; 1997.
2. Rosenblatt F. The perceptron: A probabilistic model for information storage and organization in the brain. Psychological Review.1958; 65(6):386–408. CrossRef. PMid:13602029
3. Fletcher J, Obradovic Z. Constructively learning a near-minimal neural network architecture. IEEE International Conference on Neural Networks. 1994; 1(7):204–8. CrossRef
4. Zhang Q, Wang C. Using Genetic Algorithm to Optimize Artificial Neural Network: A Case Study on Earthquake Prediction. IEEE Second International Conference on Genetic and Evolutionary Computing; 2008. p.128–31. CrossRef.
5. Weihong Z, Shunqing X. Optimization of BP neural network classifier using genetic algorithm. Intelligence Computation and Evolutionary Computation. Springer publications. 2013; 180:599–605. CrossRef
6. Rajasekaran S, Pai GAV. Neural networks, fuzzy logic and genetic algorithms: Synthesis and applications. PHI; 2003.
7. Prema KV, Agarwal NM, Krishna M, Agarwal V. Stock Market Prediction using Neuro-Genetic Model. Indian Journal of Science and Technology. 2015; 8(35).
8. Fall E, Chiang HH. Neural networks with dynamic structure using a GA-based learning method. IEEE 12th International Conference on Networking; 2015. p. 7–12.
9. Sridhar SS, Ponnavaikko M. Analysis of constructive neural network learning algorithms for pattern classification. IEEE International Advance Computing Conference (IACC); 2009.
10. Parekh R, Yang J, Honavar V. Constructive neural network architecture for pattern classification. IEEE Transactions on Neural Networks. 2000; 11(10):436–51. CrossRef. PMid:18249773
11. Sridhar SS, Ponnavaikko M. New constructive neural network architecture for pattern classification. Journal of Computer Science. 2009; 5(11):843–8. CrossRef
12. Sridhar SS, Ponnavaikko M. Improved Adaptive Learning Algorithm for Constructive Neural Networks. International

Journal of Computer and Electrical Engineering. 2011; 3(1):30–6. CrossRef

13. Koehn P. Combining genetic algorithms and neural networks: The encoding problem [M.Sc Thesis]. University of Tennessee; 1994.
14. Rohini V, Natarajan AM. Comparison of Genetic Algorithm with Particle Swarm Optimisation, Ant Colony Optimisation and Tabu Search based on University Course Scheduling System. Indian Journal of Science and Technology. 2016 Jun; 9(21). Doi no: 10.17485/ijst/2016/v9i21/85379.
15. Noersasongko E, Julfia FT, Syukur A, Purwanto, Pramunendar RA, Supriyanto C. A Tourism Arrival Forecasting using Genetic Algorithm based Neural Network. Indian Journal of Science and Technology. 2016 Jan; 9(4). CrossRef.
16. Murphy P, Aha D. Repository of machine learning databases; 1994. Available from: http://www.ics.uci.edu
17. Mendelsohn L. Preprocessing data for neural networks. Market Technologies, Wesley Chapel; 1993.