Available online www.jsaer.com

Journal of Scientific and Engineering Research, 2017, 4(9):371-376

Research Article

ISSN: 2394-2630
CODEN(USA): JSERBR

Knowledge Based Prediction through Artificial Neural Networks and Evolutionary Strategy for Power Plant Applications

K.S. Madhavan

Sr. DGM (ATE), Corp. R&D, BHEL

Abstract Artificial Intelligence (AI) as a field has grown and adapted to the changing times. Strong AI or the very fact that machines can exhibit human like intelligence still remains a controversial & distant dream. But the gap between strong AI and weak AI has been converging. A mathematical proof for this has been rendered by Schapire [2] who concludes that the notions of strong and weak learning are equivalent by training them through boosting algorithms. Noted experts in the field Kurt Gödel, Noam Chomsky and Roger Penrose [3] have said that strong AI is impossible. Well this is a matter of continuous debate with Ray Kurzweil, the futurist, pointing out that the Singularity is near. In fact present day machine learning algorithms can do a whole lot of things that were not possible in the past including occupations like nursing, teaching manifested through robotics. Genetics, Nanotechnology and Robotics are the mainstay for knowledge boosting algorithms in future. There is a large class of algorithms in AI based on computational intelligence or soft computing. Some are connectionist, simulating the human cognitive processes which can be largely grouped under the term Artificial Neural Networks (ANN), yet others are evolutionary like Evolutionary Strategy based Genetic Algorithms.

This paper is focused on prediction of power of induced draft fan in a thermal power plant based on a variety of independent features amounting to 33 which was reduced to 10 by feature selection using Evolutionary Strategy.

Keywords Artificial Neural Networks, Algorithms, Evolutionary Strategy

1. Introduction

Artificial Neural Network (ANN) methods are primarily data-driven rather than model driven. A partial understanding of the model would suffice for application in various domains. ANN has been applied to a number of applications as listed above. In each of these cases, the choice of training set and test set is customized as is the initial choice of weights. Pre-processing and post-processing of data are integral parts of the ANN application exercise. The choice of the ANN models for training again varies from case to case. A back-propagation model can be used in one case wherein the main focus is on pattern recognition and a probabilistic neural network or general regression neural network can be used in another case where the focus shifts to classification or function approximation. In fact the choice of the ANN model depends on the problem to be solved on hand.

A neural network is a massively parallel distributed processor made up of simple processing units called neurons, which has a natural propensity for storing experiential knowledge and making it available for use. It resembles the brain in two respects:

- Knowledge is acquired by the network from its environment through a learning process.
- Interneuron connection strengths, known as synaptic weights, are used to store the acquired knowledge. The weight represents the strength of the signal.

It is called a neural network because it is a network of interconnected elements. These elements were inspired from studies of biological nervous systems. In other words, neural networks are an attempt at creating machines that work in a similar way to the human brain by building these machines using components that behave like biological neurons.

The function of a neural network is to produce an output pattern when presented with an input pattern. The concept is rather abstract, so one of the operations that a neural network can be made to do is pattern classification. Pattern classification is the process of sorting patterns into one group or another.

A neural network should be able to:

- Classify patterns,
- Be small enough to be physically realizable,
- Be programmed by training, so it must have the ability to learn,
- Be able to generalize from the examples shown during training.

Neural Networks are termed parallel distributed processors. The advantage of being parallel is the potential for very high processing speeds. The advantage of distributed processing is a certain degree of tolerance. This tolerance gives the network the property called ‘graceful degradation’ [1] since it is supposed that if part of the network malfunctions, the whole system could still continue to operate albeit less than perfectly. It is said that this is preferable to complete failure.

Generally the researchers working in the neural network field can be divided into three categories: those trying to understand neurological processes by simulation of neural systems, those developing new neural algorithms to enhance the performance of artificial neural networks, and those applying variations of neural network technology to practical problems. This chapter comes under the latter category.

Schapire [2] has suggested that with the help of boosting algorithms a weak learning model can be directly converted into a strong learning model. Within the broad scope of the study of artificial intelligence (AI), research in neural networks is characterized by a particular focus on pattern recognition and pattern generation. Many neural network methods can be viewed as generalizations of classical pattern-oriented techniques in statistics [4]. Neural Network methods have had their greatest impact in problems where statistical issues dominate and where data are easily obtained.

A neural network is a directed graph. Neural network architectures are statistical in nature and make probabilistic assumptions about data. It is a combination of graphical algorithms and probability theory. Neural networks are characterized by statistical processing and modelling of data. There are three main types of statistical modelling: density estimation, classification and regression. Density estimation falls under the category of unsupervised learning, while classification and regression problems fall under the category of supervised learning. Regression is estimation of values of continuous variables.

Typical Artificial Neural Network problems conform to two types – One wherein the data is a representative set of the entire universal set of data types for that problem. In such cases training the data is of importance, test data does not have any role.

In the second class of problems the data fed into the network is only a proper subset or sample of the entire universal set of data types for that problem. Here test data has an important role to play in the final analysis.

Most real life problems conform to the latter variety. There are very few problems of the first kind.

2. Advanced studies on Neural Network Algorithms

The objective of a neural network is to perform a desired function such as pattern recognition, identification, classification etc. This is achieved by using suitable training techniques. A series of training techniques [6] have been developed during the past fifteen years to achieve the objective within shortest possible time and without any much error, when input/target pairs are presented to the network. The outputs from the layers are processed by appropriate mathematical functions depending on the nature of the problem.

2.1. Back propagation

Theoretical studies have started with linear networks: Adaline and Perceptron. But in engineering problems we need to use nonlinear separable neural networks. For such non-linear separable input/target pairs which often appear in practical applications ‘*tansig/logsig*’ transfer functions are employed for getting the outputs from the layers. A suitable network for such systems is feed forward backpropagation network. The variations in the backpropagation network are based on the various types of training techniques employed with a view to get the results quickly (fast convergence) & accurately (smallest error goal) without much error. The salient points to be considered for choosing an appropriate backpropagation neural network are:

2.1.1. The complexity of the problem to be solved

2.1.2. The number of input/output pairs (data points on the training set)

2.1.3. The number of weights and biases

2.1.4. Error goal

2.1.5. Duration for solution

2.2. Backpropagation Algorithm

Backpropagation consists of a number of iterative processes by which the first error between the outputs and the target sets is progressively reduced (a descent gradient) to the minimum by a training technique involving a learning rate and error computed. The initial parameters of weights and biases of the network are accordingly modified to get the desired goal rapidly and accurately. The backpropagation algorithm is

$$X_{k+1} = X_k - \alpha_k\, g_k$$

Where X_k is the current weights and bias vector, α_k is the learning rate and g_k is the first derivative of the gradient of the error.

2.3. Neural Network Training

There are two modes of training: Incremental mode and Batch mode

In incremental mode when input/target pairs are presented, each input and target is incrementally trained at every iteration (initial weights and biases are changed to new weights and biases) till the goal is achieved. In batch mode the weights and biases are updated only after the entire training set (all input/target pairs) has been applied, the gradients of each are added together and then applied to new weights and biases. Generally a momentum parameter is also applied for faster convergence (gradient). This takes into account before changing weights and biases, sum of a fraction of the last weight change and the new weight change suggested by back propagation. *Learngd* & *Learngdm* respectively are such techniques in the incremental mode. *Traingd* & *Traingdm* respectively are such techniques in the batch mode used in MATLAB software. This software uses a variety of fast learning algorithms that give quick and accurate results for solving data related problems on hand.

2.4. Learning Algorithms

2.4.1. Variable learning rate algorithms: In these α_k instead of being constant throughout is varied during the process suitably to ensure stability of the network (no oscillations) and also rapid convergence. *Traingda* & *Traingdx* (with momentum) are of the variable learning rate techniques of backpropagation and our calculations of all backpropagation networks are made in the batch mode.

2.4.2. Resilient backpropagation (Trainrp): This multilayer network using sigmoid transfer function is much faster than the previous one requiring a modest increase in memory requirements (computations required to make weight & bias changes) and removes the harmful effects of the partial derivative at the ends of the sigmoid function where the slopes are very small by using the sign of the derivative rather than the magnitude.

2.4.3. Conjugate gradient algorithms: In the above types, the weights update, for each iteration, is made by the step size in the negative gradient direction by learning rate. In the conjugate gradient algorithms, this step size is modified by a search function at every iteration such that the goal is reached as early as possible within a few iterations. This large size step is obtained by combining the new steepest descent direction with the previous one.

$$p_k = -g_k + \beta_k\, p_{k-1;}$$

For all conjugate gradient algorithms the search direction is periodically reset. The standard reset point is when

Number of iterations = number of network parameters

The Fletcher-Reeves update and Polak-Ribiere update are versions of conjugate gradient algorithms and are distinguished by the manner in which the constant β_k is computed.

Fletcher-Reeves update (cgf) is much faster than variable learning rate algorithms & resilient backpropagation but requires a little more storage as computations are more but suffers from the fact that the results may vary from one problem to another.

Polak-Ribiere update (cgp) (is similar to the above cgf). The value of β_k is different from that of Fletcher-Reeves update.

Powell-Beale restarts (cgb) the resetting technique is changed from that of Fletcher-Reeves update & Polak-Ribiere update above. It is known for its efficiency and very rapid convergence. It requires a little more storage than that for Fletcher-Reeves update & Polak-Ribiere update.

Scaled Conjugate Gradient (scg): This technique combines the model trust region approach of Levenberg Marquardt algorithm with conjugate gradient approach. No line search is made resulting in slower convergence but less number of computations & hence the storage requirements are reduced.

2.4.4. Quasi-Newton algorithms: are an alternative to conjugate gradient. It provides for faster optimization of results in backpropagation: Broyden Fletcher Goldfarb Shanno (BFGS) algorithm; One Step Secant (OSS) algorithm

BFGS: faster but requires more computation

OSS: similar to BFGS but requires less storage and computation

2.4.5. Levenberg Marquardt algorithm (LM): It is designed to approach second order training speed without having to compute Hessian but modified to use Jacobian matrix. As such very few epochs (passes) are required, large network problems could be employed.

2.4.6. Reduced Memory LM: is a particular case of LM where storage requirements that may be required in LM for certain large network problems will be reduced by splitting Jacobian matrix into two half Jacobian matrices and thus has a reduced memory requirement.

2.5. Speed & Memory requirements

Based on the neural network studies made on typical problems, the following order of techniques is recommended for solution of large complex problems: Levenberg Marquardt, BFGS Quasi-Newton, Powell-Beale conjugate gradient, Resilient Backpropagation (Rprop).

2.6. Generalisation

For good and efficient neural networks, an important characteristic that it should have is *generalisation*. Generalisation is the characteristic by which it is ensured that when the input/target training set is trained to give the desired error goal, it should give the same error when it is presented with a new test set of input/targets. In most of the cases when the test set is presented for training, the error is found to be large. There are two methods for improving this generalisation: Regularisation and Early Stopping

Regularisation is obtained by modifying the performance function by adding a term consisting of the mean of the sum of the squares of network weights & biases.

$$\text{Msereg} = \gamma \text{ mse} + (1 - \gamma) \text{ msw}$$

Early Stopping is another method for improving generalisation. When the test set error starts increasing from that obtained during training set, the training is stopped.

3. Case Study of Prediction of Induced Draft Fan Power

The Induced Draft (ID) Fan power output is directly or indirectly dependent on 33 input features including Inlet flow rate, Fan total pressure, Fan efficiency, Fan working speed etc. and output being Fan shaft power.

Firstly the selected features were tested with chi squared test and found that outliers were removed. This test sieved out the extreme values and adopted only relevant values for prediction.

All 33 input features need not be important in predicting the Output Power of the ID Fan. There is relative importance with some features having more importance than the others. Feature selection is an important exercise to be done in order to arrive at these selective independent features. This reduces computational time, complexity of the problem and increases accuracy of prediction.

In our case study we have computed FSTAT and the corresponding probability value for the 33 independent features and found out 10 relatively important features for the purpose of our prediction exercise. The results of the study are seen in the figure 1 below:

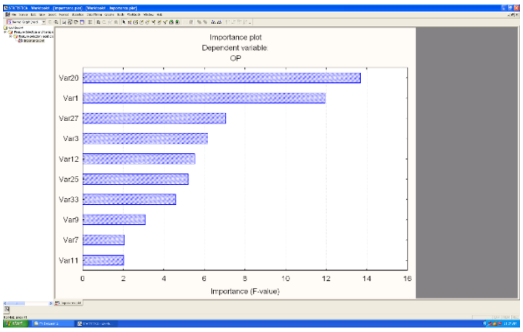
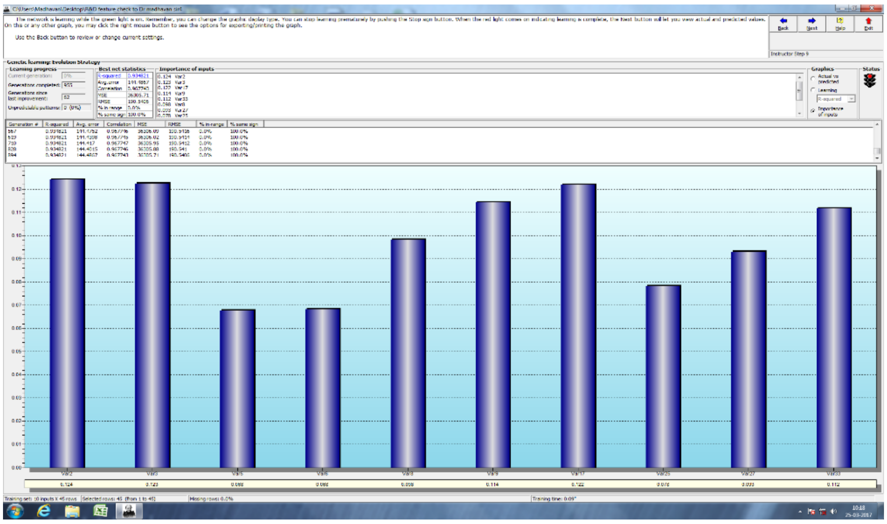


Figure 1: The ten best predictors after applying FSTAT and corresponding probability

Feature	F-value	p-value
Var20	13.68246	0.000000
Var1	11.93671	0.000000
Var27	7.04848	0.000023
Var3	6.15326	0.000053
Var12	5.52016	0.000210
Var25	5.20315	0.000232
Var33	4.59401	0.000895
Var9	3.08211	0.009446
Var7	2.04098	0.069011
Var11	2.02096	0.086663

In order to validate the feature selection done by the above approach, we have used a genetic algorithm based on evolutionary strategy [6] iteratively in order to arrive at the important features. This is a novel technique and combines the power of evolutionary computation and analysis of variance to give an output that will be more realistic for the purpose of validation. The technique used was evolutionary strategy algorithm that would maximize the coefficient of multiple determination and minimize number of unpredictable patterns with a breeding pool size and initialization that can be set based on problem complexity.

With the above set options the genetic algorithm was run for several generations till it stabilized. The trained network was set such that the top 27 features out of 33 were selected as features based on relative importance for purpose of prediction of output power of ID fan. Then the process was repeated iteratively to select the top 25 features, 15 and 10 respectively. These 10 features selected were treated as the final list of independent features on which the ID fan output power was dependent on. This list was plotted and is shown in the figure 2 below:


C:\Users\Madhavan\Desktop\R&D feature check to Dr madhavan sir\

The network is learning while the green light is on. Remember, you can change the graphic display type. You can stop learning prematurely by pushing the Stop sign button. When the red light comes on indicating learning is complete, the Next button will let you view actual and predicted values. On this or any other graph, you may click the right mouse button to see the options for exporting/printing the graph.

Use the Back button to review or change current settings.

Genetic learning: Evolution Strategy

Learning progress

Current generation: 0%
Generations completed: 955
Generations since last improvement: 62
Unpredictable patterns: 0 (0%)

Best net statistics

R-squared	0.934821
Avg. error	144.4867
Correlation	0.967743
MSE	36305.71
RMSE	190.5406
% in range	0.0%
% same sign	100.0%

Importance of inputs

0.124	Var2
0.123	Var3
0.122	Var17
0.114	Var9
0.112	Var33
0.098	Var8
0.093	Var27
0.078	Var25

Generation #	R-squared	Avg. error	Correlation	MSE	RMSE	% in-range	% same sign
567	0.934821	144.4752	0.967746	36306.09	190.5436	0.0%	100.0%
619	0.934821	144.4308	0.967745	36306.02	190.5434	0.0%	100.0%
710	0.934821	144.417	0.967747	36305.95	190.5412	0.0%	100.0%
828	0.934821	144.4015	0.967746	36305.88	190.541	0.0%	100.0%
894	0.934821	144.4867	0.967743	36305.71	190.5406	0.0%	100.0%

Var2	Var3	Var5	Var6	Var8	Var9	Var17	Var25	Var27	Var33
0.124	0.123	0.068	0.068	0.098	0.114	0.122	0.078	0.093	0.112

Training set: 10 inputs X 45 rows | Selected rows: 45 (from 1 to 45) | Missing rows: 0.0% | Training time: 0:09"

Figure 2: The Ten Best Predictors after applying evolutionary strategy optimisation iteratively

The comparison shows that five features are common to both the feature selection approaches. This intersecting set may be considered as the final feature selection set for ID fan power output prediction.

But in our experiment 10 features were considered and trained through a cascade correlation neural network model [7] in order to predict the output ID fan power. The results were achieved to 98.6% accuracy.

4. Conclusion

Artificial Neural Network is good for knowledge based prediction. But feature selection of input parameters is a must when we have to choose between many existing independent features. This improves computational power and accuracy of prediction. This ensures that the right features are taken for prediction. The evolutionary strategy based feature selection, considered in this chapter is unique and novel in approach and has been found to give good results during ANN prediction. In this paper a power plant application of predicting the output power of an ID fan has been considered with appropriate feature selection and prediction through cascade correlation for a desired accuracy.

Acknowledgements

The author is grateful to BHEL management for according permission to publish this work.

References

[1]. B. Yegnanarayana, Artificial Neural Networks, Prentice-Hall, 2004.

[2]. Schapire, R.E., “The strength of weak learnability”, Machine Learning, vol.5, pp 197-227, 1990.

[3]. Roger Penrose, “The Emperor's New Mind: Concerning Computers, Minds and The Laws of Physics”, Oxford University Press, 1989

[4]. Afifi, A.A, and Clark,V. Computer Aided Multivariate Analysis, Fourth Edition, Chapman & Hall, 2003.

[5]. Borovikov, I.P and Borovikov, V.P. STATISTICA: Data Preparation and Analysis. Moskow : Filini, 1998.

[6]. Howard Demuth, Mark Beale, “Neural Network Toolbox”, 1998.

[7]. Michalewicz, Z., "Genetic Algorithms + Data Structures = Evolution Programs", Second, Extended Edition, Springer-Verlag, New York, NY, 1992, chapter 8, Evolution Strategies and Other Methods.

[8]. Ward Systems Group, Neuroshell Predictor Manual, 2016.