International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-9 Issue-2, December, 2019

Constructive Neural Network: A Framework

Jaswinder Kaur, Neha Gupta

Abstract*: In this paper, two techniques for construction of feedforward neural network are being reviewed: pruning neural network algorithms and constructive neural network algorithms. In pruning method, training starts with a larger than required network and subsequently delete the redundant hidden nodes and redundant weights till there is a satisfactory solution. In the constructive method, training of the network starts with minimum structure and then according to some predefined rule some more layers of neurons are added. A number of major issues are discussed that can be considered while constructing a constructive neural network i.e. how to select network architecture, network growing strategy, weight freezing, optimization technique, activation function and stoppage criteria

Keywords: Neural networks; Pruning algorithm; Constructive algorithm; Optimization technique and Activation function.

I. INTRODUCTION

Neural network research has come a long way and it has established well in the field of modeling linear and nonlinear data. There is lot of demand of adaptive techniques for solving neural network problems. Neural networks are used in many different applications like function approximation, pattern recognition, image processing, speech recognition, classification and other modeling tasks [1]. Multilayer feedforward neural networks are the most suitable models for solving problems of nonlinear mapping [2]. Neural network training consists of parametric learning and structural learning [3]. Neural networks are used in many applications like extraction of knowledge, forecasting (bankruptcy, weather), healthcare (clinical diagnosis, image analysis), communication, robotics, data processing and compression, approximation of function. This network learns adaptively, performs real time operations and is fault tolerant.

The multilayer feedforward neural network (FNNs) is widely used in many applications. The popularity of FNNs has been traced to the structure flexibility, capability of good approximation, and availability of large number of algorithms for training. The design of FNN for a given task comprises of many components / parameters, e.g. network architecture, activation function, learning algorithm and the other training parameters. The generalization performance of network and convergence time of network learning in FNNs depend on

how its constituents are selected like architecture of network i.e. number of nodes in hidden layer and topology used for connections between nodes, activation function for each node and training parameters i.e. initial weights, learning rate etc.. The conventional FNN requires the architecture of network to be specified before the training starts [4]. In general, the designer of the network defines the network architecture by trial and error method [5]. While developing the neural network the number of elements needed for processing is not known in advance or found by trial and error method but are found while finding the solution of the problem. The generalization and training time of neural network are affected by the size of the network and the topology chosen for building the network. If the network architecture chosen is not appropriate, then the network can be underfitting the problem or overfitting the problem. The number of trainable parameters should be enough in number and should be able to capture from the training information the mapping function. For a given problem it is difficult to find the best network topology. A small network might not be able to learn the problem properly whereas a large network may overfit the data used for training resulting in poor generalization of the problem performance. From the desired output of the problem we can determine if the problem is a regression problem or classification problem. Regression problems involve approximation of continuous valued target function. It consists of discrete-continuous and continuous-continuous input-output mappings [6].

II. CONSTRUCTIVE NEURAL NETWORK

Nowadays, adaptive techniques are required for solving problems. In adaptive structure neural network the structure of the network is adapted while training the network according the problem [7]. The adaptive architecture algorithms are of two types: pruning neural network and constructive neural network. The pruning algorithm starts with network architecture which is larger than required and then the redundant nodes of hidden layer and weights are deleted. This process continues until we find a satisfactory solution [8]. In constructive neural network (CoNN) an opposite approach is followed in which network building starts with minimum architecture elements and then nodes in the hidden layer are added one at time. Advantages of constructive approach over pruning approach are, in constructive approach it is easy to build the initial network, it always finds a smaller network solution whereas, in pruning approach the starting size of the network is very difficult to decide. As there is no method available using which the network architecture can be decided, there is need for an algorithm which can find appropriate network architecture for the problem.

Revised Manuscript Received on December 30, 2019.

* Correspondence Author

Jaswinder Kaur, School of Engineering & Technology, Ansal University, Gurgaon, India. E-mail:jasukaur@rediffmail.com

Neha Gupta, School of Engineering & Technology, Ansal University, Gurgaon, India. E-mail: nehagupta@ansaluniversity.edu.in

III. DESIGNING THE NETWORK

This section presents a framework for designing a constructive neural network for regression problems. CoNNs start with a small network and while learning the network architecture grows until the solution of the problem is found.

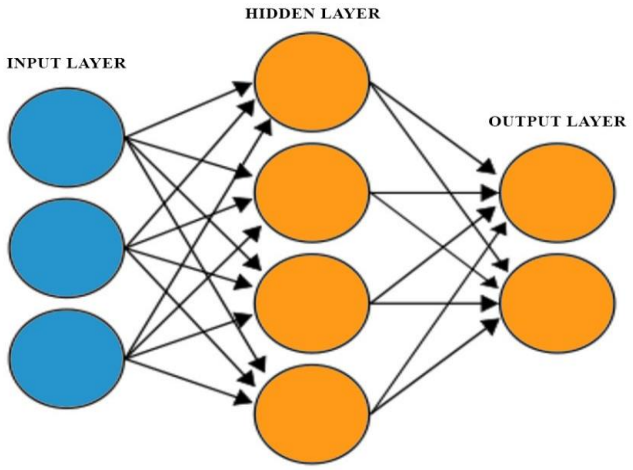

Fig 1: neural network with three input units, four hidden units and two output units.

There are six main decision issues involved in designing algorithm for problems that constructs FNN incrementally. The issues are:

- Minimal network architecture - How to select the minimum network architecture?
- Strategy for growing the network – How the new nodes are to be connected in the network?
- Weights freezing technique – While training the network do we need to train the whole network or only the node which is newly added?
- Optimization techniques – Out of the various optimization methods which optimization method can be used for adapting the weights?
- Activation function - From the various activation functions available it is to be decided which activation function can be used for nodes of hidden layer and for nodes of output layers.
- Training stoppage criteria - While training the network it is to be decided when to stop training the network.

The ability of generalization and time required for training a constructive neural network for regression problems depend on points discussed above. It adds hidden nodes one by one and trains them incrementally. All these points emphasize adaptation according to architecture and functions. The next section presents these points in detail [9].

A. Minimal Network Architecture

Depending on the given problem, the number of input nodes and the number of output nodes is decided. These numbers are not changed during the process of architectural adaptation. Various methods can be used like pruning or destructive method, constructive method, regularization and constructive-pruning hybrid method. For constructive method, let's focus on Dynamic node Creation algorithm (DNC) or Cascade correlation algorithm (CCA) for selection of network architecture.

DNC and its variants start building the network with single hidden node or some hidden nodes in a single hidden layer FNN. It adds hidden nodes to the same layer.

CCA and its variants start the network with direct input-output connections without any hidden node. CCA adds only one node at a time to a different layer each time.

B. Strategy for Growing the Network

CoNN starts the network with a minimum architecture and then grows the network architecture while training the network according to the problem. CoNN find during training of the network, connection topology and number of nodes in the hidden layer. For single layer feedforward neural network the hidden node is added in single layer. For multilayer feedforward neural network the hidden node is added in different layers. In the cascade network architecture there is only one node in each hidden layer.

These two pre-specified network-growing strategies that construct SLFNN and cascade architecture are discussed below.

In Dynamic Node Creation (DNC) [10] algorithm construction of the network starts with a small network containing small number of nodes in hidden layer and then adding one hidden node at a time using backpropogation method till a solution of the problem is found [11]. All the connections between layers are feedforward. In the network, nodes in the input layer are connected to the nodes of hidden layer and nodes in the hidden layer are connected to the output layer [12].

In Cascade Correlation new hidden unit is first trained and then added one at a time starting with a small network. There are two steps in Cascade Correlation algorithm. First step consists of building the network with adding one hidden node at one time. Once the hidden node is added in the network it can't be changed. In this new hidden unit are first trained and then added one at a time. Second step consists of creating the new node and installing the hidden node using learning algorithm [13].

C. Weight Freezing Technique

There are a number of ways in which the hidden node can be trained. The ways of training the network can be classified into four categories. The first, in this the whole network is trained after adding a new node in the hidden layer of the network. It is called no freezing method. The second, in this method only training of new hidden node weights is done and all other weights remain unchanged [14]. The third, in this method the weights of input connection of trained hidden node and output connection of trained hidden node are freezed permanently. And training of input weights and output weights is done using one objective function simultaneously. The fourth, in this the newly added nodes are trained with higher learning rate and the earlier or older nodes of the hidden layer are trained with lower learning rate. DNC and its variants use the first approach; as a consequence, the resulting algorithm is computational expensive and has slower convergence. Although second approach is very computational efficient, it converges very quickly, and also helps to avoid the problem of moving-target.

CCA and the variants of CCA use two phase training. In the first phase, only the input weights which are connected to a current hidden node are trained. After inserting this node in the current network, these weights are frozen and then only the weights which are connected to the output nodes are trained again.

The requirement of two stage training is necessary for an algorithm having two cost functions. There is an additional feature of CCA in which several candidate nodes are trained in parallel and out of them the node whose performance is best is selected and added to the network currently being built. This additional feature is utilized in INCA. This feature is not utilized here for the sake of simplicity, as it is very expensive computationally.

Incremental node creation algorithm (INCA) the network building starts with one hidden node and training one node at a time incrementally. Two variants of INCA are cascade incremental node creation algorithm and flat incremental cascade node creation algorithm. In cascade INCA, each node is added in a new hidden layer. In flat INCA, each node is added to the same single hidden layer. In both the variants updating of input connection weights and output connection weights is done and keeping rest network freezed.

In the Cascade network algorithm using progressive RPROP algorithm (CasPer) [15] a very small step-size is used for the inputs of hidden nodes, instead of freezing entirely while the connections to the output nodes use a larger step-size in the second phase. This algorithm is constructed in the same way as the CasCor network. In CasPer algorithm building of network starts with one node in hidden layer and then one by one hidden node are added in the network. After adding a hidden node to network, the whole network is trained using RPROP (Resilient backpropogation) algorithm [16].

D. Optimization Technique

For adapting the weights of the network in CoNN any algorithm based on local optimization method can be used for fixed size FNN [17]. Neural network needs to be trained each time its architecture is modified whereas feedforward neural network with fixed size needs to be trained once only. There are various non-linear optimization methods like Newton method, Conjugate Gradient, Quasi-Newton (BFGS) method, Gauss-Newton method, Levenberg Marquardt method, Quickprop, RPROP method. The exact computational requirement depends on the method of optimization. For training a network with X_n weights using gradient descent method $O(X_n)$ operations are required, using quasi-Newton method $O(X_n^2)$ operations are required, using Levenberg Marquardt method $O(X_n^3)$ operations are required, using Gauss-Newton method $O(X_n^3)$ operations are required and using Newton method $O(X_n^3)$ operations are required in each iteration [18, 19].

E. Activation Function

Activation function is used for calculating the output. It is selected to fulfill some specific requirements of the problem that are tried to solve. There can be linear or non-linear activation functions. Activation functions can be of following types: Ramp function, Identity function, Bipolar step function, Binary step function and Sigmoidal function like Binary or Bipolar sigmoidal function. Some of the activation functions are Hard Limit, Symmetrical Hard Limit, Linear, Saturating Linear, Symmetric Saturating Linear, Log-Sigmoid, Hyperbolic Tangent Sigmoid, Positive Linear and Competitive function. Mostly used activation function is logistic sigmoid function with a range of [-1, 1].

$$f(a) = \frac{1}{1 + e^{-a}} \quad (1)$$

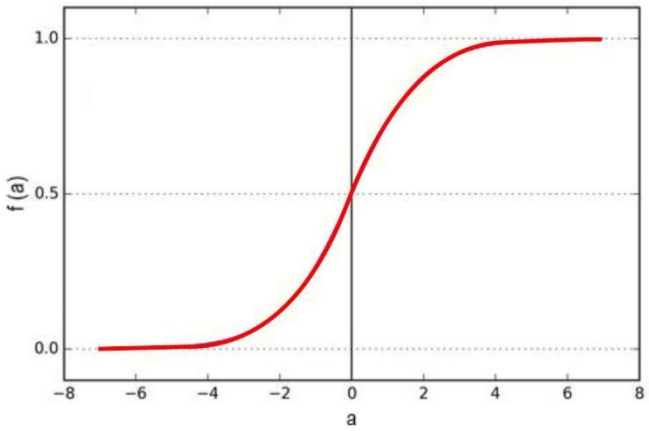

Fig 2: Logistic sigmoid function.

F. Training Stoppage Criteria

The standard method is followed for adding a new hidden node in all CoNNs. In this method if the training error does not come down to particular value in the given amount of time then, a new node is added to the hidden layer. It is important to find out when to stop adding node to the network. It can be done when the network learns the desired mapping to a user specified precision. Once the desired level of performance is reached, it is important to prevent further node growth. Various classes of stopping criteria can be used to find out when to stop training of the network. Stopping criterion should be chosen that leads to lowest possible generalization error. The ratio of performance to price should be the best. Early stopping techniques are used for halting network construction.

IV. CONCLUSION

Multilayer feedforward neural networks are the most powerful and widely used in many applications. Out of the two adaptive techniques constructive method is better than pruning method. The six major issues discussed above can be considered while constructing constructive neural networks.

REFERENCES

1. Selot, Smita, Neeta Trpathi, and A. S. Zadgaonkar. "Knowledge Representation in pAninI Framework Using Neural Network Model." and many more..., for more details click at http://www. bvicam. ac. in/BIJIT/indexing. asp (2013): 537.
2. Lehtokangas, Mikko. "On weight initialization in cascade-correlation learning." Neural Networks, 1999. IJCNN'99. International Joint Conference on. Vol. 3. IEEE, 1999.
3. Lahnajärvi, Jani JT, Mikko I. Lehtokangas, and Jukka PP Saarinen. "Estimating movements of a robotic manipulator by hybrid constructive neural networks." Neurocomputing 56 (2004): 345-363.
4. Verma, A. K., R. Anil, and Om Prakash Jain. "Fuzzy Logic Based Revised Defect Rating for Software Lifecycle Performance Prediction Using GMR." Bharati Vidyapeeth's Institute of Computer Applications and Management (2009): 1.
5. Chande, Swati V., and Madhavi Sinha. "Genetic algorithm: a versatile optimization tool." BIJIT-BVICAM's International Journal of Information Technology 1.1 (2009): 7-12.

6. Lahnajärvi, Jani JT, Mikko I. Lehtokangas, and Jukka PP Saarinen. "Evaluation of constructive neural networks with cascaded architectures." Neurocomputing 48.1 (2002): 573-607.
7. Khan, W. A., Chung, S. H., Ma, H. L., Liu, S. Q., & Chan, C. Y. (2019). A novel self-organizing constructive neural network for estimating aircraft trip fuel consumption. *Transportation Research Part E: Logistics and Transportation Review*, *132*, 72-96.
8. Daniel, W. B., & Yeung, E. (2019). A constructive approach for one-shot training of neural networks using hypercube-based topological coverings. *arXiv preprint arXiv:1901.02878*.
9. Setiono, Rudy, and Lucas Chi Kwong Hui. "Some n-bit parity problems are solvable by feedforward networks with less than n hidden units." Neural Networks, 1993. IJCNN'93-Nagoya. Proceedings of 1993 International Joint Conference on. Vol. 1. IEEE, 1993.
10. Ash, Timur. "Dynamic node creation in backpropagation networks." Connection Science 1.4 (1989): 365-375.
11. Wu, Z. Q., Jianmin Jiang, and Y. H. Peng. "Computational Intelligence on Medical Imaging with Artificial Neural Networks in." Computational intelligence in medical imaging techniques and applications (2009).
12. Fahlman, Scott E., and Christian Lebiere. "The cascade-correlation learning architecture." (1989).
13. Parekh, Rajesh, Jihoon Yang, and Vasant Honavar. "Constructive neural-network learning algorithms for pattern classification." IEEE Transactions on neural networks 11.2 (2000): 436-451.
14. Thakur, Ghanshyam Singh, and R. C. Jain. "NFCKE: New Framework for Document Classification and Knowledge Extraction." Bharati Vidyapeeth's Institute of Computer Applications and Management (2009): 55.
15. Treadgold, Nick K., and Tamás D. Gedeon. "Extending and benchmarking the CasPer algorithm." Australian Joint Conference on Artificial Intelligence. Springer Berlin Heidelberg, 1997.
16. Gedeon, T. D., and N. K. Treadgold. "Extracting meaning from cascade networks." Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on. Vol. 4. IEEE, 1997.
17. Islam, Md Monirul, et al. "A new constructive algorithm for architectural and functional adaptation of artificial neural networks." IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 39.6 (2009): 1590-1605.
18. Kwok, Tin-Yau, and Dit-Yan Yeung. "Constructive algorithms for structure learning in feedforward neural networks for regression problems." IEEE Transactions on Neural Networks 8.3 (1997): 630-645.
19. Prechelt, Lutz. "Early stopping-but when?." Neural Networks: Tricks of the trade. Springer Berlin Heidelberg, 1998. 55-69.

AUTHORS PROFILE

Jaswinder Kaur received her B.Sc. (Computer Science) degree from Kurukshetra University, Kurukshetra and MCA degree from Kurukshetra University, Kurukshetra. She has teaching experience of 5 years. Her areas of interest are Neural Networks, Web Technology, Database Systems and programming languages. Her current research interest is in Artificial Intelligence.

Neha Gupta has teaching experience of about 17 years. Her areas of interest are renewable energy fuels, power system and ANN. She has published 2 patents in Indian Journal of patents .She has presented and published many technical papers in various reputed International Journals and Conferences at National and International level. She is also an author of 2 books.