Int. J. Nonlinear Anal. Appl. 15 (2024) 5, 177–187
ISSN: 2008-6822 (electronic)
http://dx.doi.org/10.22075/ijnaa.2022.29622.4208

Automatic fault diagnosis of computer networks based on a combination BP neural network and fuzzy logic

Elham Bideh[a], Mohammadreza Fadavi Amiri[a], Javad Vahidi[b,*], Majid Iranmanesh[c]

[a]*Department of Computer Engineering, Shomal University, Amol, Iran*

[b]*Department of Computer Science, Iran University of Science and Technology, Tehran, Iran*

[c]*Department of Mathematics, Semnan University, Semnan, Iran*

(Communicated by Madjid Eshaghi Gordji)

Abstract

Today, computer network fault diagnosis is one of the key challenges experts are facing in the field of computer networks. Therefore, achieving an automatic diagnosis system which is based on artificial intelligence methods and is able to diagnose faults with maximum accuracy and speed is of high importance. One of the methods which is studied and utilized up to now is artificial neural networks with a back propagation algorithm while using neural networks with a back propagation algorithm has two main challenges in front. The first challenge is related to the backpropagation learning type as it is a supervised learning requiring inductive knowledge driven from previous conditions. The second challenge is the long time required for training such a neural network. In this work, combining neural networks with a backpropagation algorithm and fuzzy logic is applied as a method for confronting these challenges. The result of this study shows that fuzzy clustering is able to provide the inductive knowledge required for backpropagation learning by determining the membership degree of training samples to different clusters of network faults. Also, according to the simulations taken place, implementing a fuzzy controller in determining the learning rate in each backpropagation iteration has resulted in successful outcomes. Thus, the learning speed of this algorithm has been increased in comparison to the constant learning rate mode resulted in reducing the training time duration of this neural networks.

Keywords: Computer Networks Fault Diagnosis, Artificial Neural Networks, Back Propagation Algorithm, Fuzzy Clustering, Fuzzy Controller
2020 MSC: 68T05

1 Introduction

Computer networks are always expanding and growing in two aspects. First, in terms of size, and second, in terms of usage and people's dependence on them in daily life [14]. This is due to the rapid development and growth of the Internet and communication technologies [1] and raises the sensitivity and importance of managing and maintaining computer networks in dealing with any challenge, fault, attack and risk. Therefore, obtaining an automatic system that notifies the network manager when faults occur by automatically detecting computer network faults and has

*Corresponding author
Email addresses: e.bideh@outlook.com (Elham Bideh), fadavi@shomal.ac.ir (Mohammadreza Fadavi Amiri), jvahidi@iust.ac.ir (Javad Vahidi), iranmanesh.majid@alum.semnan.ac.ir (Majid Iranmanesh)

Received: August 2022 *Accepted:* September 2022

maximum speed and accuracy in this regard is certainly important. One of the proposed methods in the field of fault detection in various areas is the use of Artificial Neural Networks (ANNs). This method achieves strong performance on irrelevant input data, noise and non-linear data [16], and this research intends to use this method with a different approach.

Artificial neural networks are systems that process information non-linearly and are made up of interconnected primary processor components called neurons. The pattern of functioning of the ANNs is inspired by the nervous system of living organisms [23]. Having a variety of capabilities including self-learning, compatibility, real-time function, fault tolerance, ease of execution and low cost, makes ANNs suitable for a diversity of applications [17]. Time-delay neural networks, high-order nets, recurrent nets and Back Propagation (BP) are some of the top neural patterns [12].

Inspired by the human beings' ability of learning from their mistakes, learning from mistakes in neural networks also could be done using feedbacks given to the input patterns. These feedbacks will be used for rebuilding the input pattern and minimizing the errors; resulting increase in the performance efficiency of the neural network. Due to the complexity of this method building these kinds of neural networks is very hard. These networks are known as auto-associative neural networks. As the name suggests, such types of networks use BP algorithm [12].

Among the different neural network algorithms, BP algorithm to recognize complex patterns are appropriate and if the access to data sets adequate with defined input and target, it is able for optimal learning in order to use in automated computer network fault diagnosis software. But there are some disadvantages in this algorithm that can be improved by combining this type of neural network algorithm with other methods of artificial intelligence.

This paper focuses on combining neural network with BP algorithm and fuzzy logic. Fuzzy logic reflects the real world and specialized human behaviors. For this reason, it is expected that in combination with neural networks modeled based on the human nervous system, it is perform optimally and significantly reduce the weaknesses in neural network algorithms.

The remainder of paper is as follows. Section 2 introduces basic concepts. Section 3 will explain increasing the speed of neural network with BP algorithm by fuzzy logic controller. Section 4 contains the results and Implementation process of the proposed approach and the conclusion and suggestions will be discussed in Section 5.

2 Basic Concepts

2.1 Fuzzy Clustering

Learning the classification process by a neural network with BP algorithm is a type of supervised learning [13] that requires sufficient inductive knowledge including input complete features as well as output result from combining each set of input features (target). In some cases, there is no possibility of providing a target data set consisting of various combinations of input features. So, we need a self-learning part to predict the required output states. This self-learning unit operates unsupervised and classifies the various combinations of features, each called a training example, into separate clusters. Common results fall into identical clusters and the resulting clusters are used as targets of training samples in BP neural network training. Different clustering methods may be used for this initial clustering. There are two types of clustering algorithms based on the relationship between each data (sample) and each cluster: Crisp clustering and fuzzy clustering. In crisp clustering, each data definitely belongs entirely to one cluster. But, this type of clustering is not suitable for data that is inherently fuzzy or suspicious and cannot be fully assigned to a cluster [3]. We intend to use fuzzy clustering in this paper. When it comes to fuzzy clustering, it does not fully seize the sample winning cluster. Each sample belongs to all clusters with a specific membership degree, but the degree of belonging to each cluster is different. For each sample, the membership degree to each cluster can be between 0.0 and 1.0. The value 0.0 illustrates the least membership value, while 1.0 shows the highest membership value. This is exactly the fuzzy looking at computer network faults. The general framework of the proposed fault diagnosis system is indicated in Fig. 1.

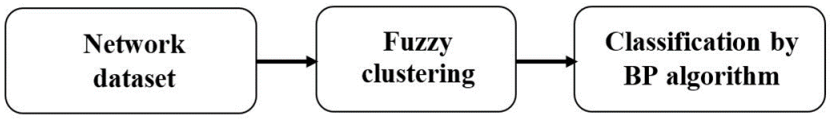

Figure 1: General framework of the proposed fault diagnosis system

The basic Fuzzy C-Means (FCM) clustering algorithm is the base of fuzzy clustering analysis. This algorithm was originally proposed by Bezdek [5] and [6]. FCM assigns data sets to different fuzzy clusters using fuzzy theory based

on specific criteria [9]. The FCM algorithm has been used in image processing, pattern recognition, cluster analysis, etc. [25] and in this paper, it is used for clustering network faults.

Any clustering algorithm needs to use a mechanism for measuring distance between objects. To make decisions on clustering, the algorithm uses distance measuring. Generally Euclidean Distance measure is used to find similarity between objects. FCM generates features vectors and keeps each feature vector into every cluster with a value associated with them. This value is known as fuzzy truth value which is between 0 and 1.

The proposed framework for network faults clustering has four phases. In the first phase, a membership matrix is initialized. In the second phase, fuzzy cluster centroid is computed. In the third phase, termination condition is verified. Fuzzy membership is updated in the fourth phase. Fig. 2. Shows the proposed framework for clustering network faults [18].

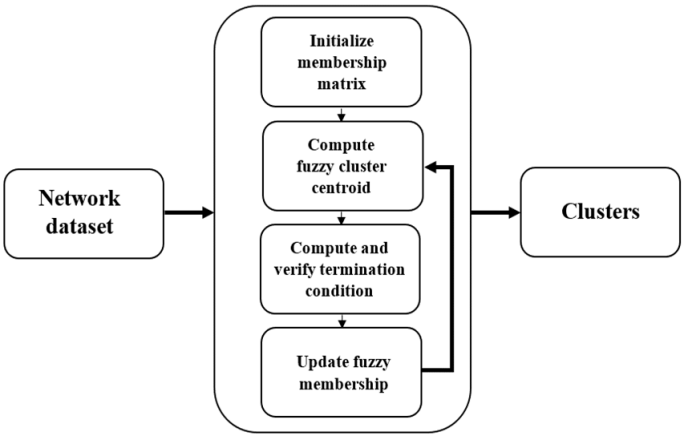

Figure 2: Proposed framework for clustering network faults [18]

As depicted in Fig. 2, the system takes network dataset as input and generates clusters that represent various network faults. The objective of FCM algorithm is to obtain fuzzy C partition $F = \{\mathrm{F_1}, \mathrm{F_2}, \mathrm{F_3} \ldots \mathrm{F_n}\}$ for given dataset $X = \{\mathrm{X_1}, \mathrm{X_2}, \mathrm{X_3} \ldots \mathrm{X_n}\}$ and number of clusters denoted by "c". It has to be achieved by minimizing J_m as follows [18]:

$$J_{\mathrm{m}}(\mu, V : X) = \Sigma_{i=1}^{c} \Sigma_{j=1}^{n} (\mu_{\mathrm{ij}})^m \|X_{\mathrm{j}} - V_{\mathrm{i}}\|^2 \tag{1}$$

Where μ_{ij} indicates the membership value of data point X_j in the i_{th} cluster. m is any real number that is greater than 1 and known as the fuzzifier. The center of i_{th} cluster is V_i, whereas $\|.\|$ defines the Euclidean Distance.

As per the proposed framework illustrated in Fig. 2, initialization of membership value is done as follows:

$$\Sigma_{i=1}^{c} \mu_{\mathrm{ij}} = 1 \tag{2}$$

First, the membership value and centroids of clusters are randomly initialized considering some limitations. Then the computation of fuzzy cluster centroid is done as follows:

$$V_{\mathrm{i}} = \frac{(\Sigma_{j=1}^{n} (\mu_{\mathrm{ij}})^m X_{\mathrm{j}})}{(\mu_{\mathrm{ij}})^m} \tag{3}$$

Then, the equation (1) is used for computing the value of $J_m(U,V)$ and determining the criterion function based on a threshold. Next, the fuzzy membership is updated as follows and this process is repeated until convergence:

$$\mu_{\mathrm{ij}} = [\Sigma_{k=1}^{c} (\|X_{\mathrm{j}} - V_{\mathrm{i}}\|^2 / \|X_{\mathrm{j}} - V_{\mathrm{k}}\|^2)^{1/(m-1)}]^{-1} \tag{4}$$

After the fuzzy clustering, the results are used as the training target for the set of training samples, which are the inputs of the BP algorithm.

2.2 Neural Network with BP Algorithm

Neural network with BP algorithm is able to promote the model that makes it good for diagnosing the faults. To use BP algorithm in computer networks fault diagnosis, the following steps should be considered [26]:

• Determining an appropriate and acceptable structure and model for the neural network. In particular, the number of neurons in the middle layer of the neural network plays a major role in structure and performance of network.

• Providing a suitable data set and specifying parts of the data set as the training set as well as the test set. The training set will be used in the neural network training process while the test set is used to validate the trained neural network. The training set should be representative of all the different scenarios of computer networks faults.

• Based on the training set and the results of the validation, the computer networks fault diagnosis neural network will be trained.

In the training phase of a typical BP algorithm for a multilayer network, the supervisor gives the input data to the network and evaluates the actual output of the network. The difference, or error, between the expected output (T_i) and the actual output of the network (O_i) is calculated by equation (5) and is used through equations (6) and (7) to update the weight of neurons in the neural network [20]).

$$E_{\mathrm{r}} = (T_{\mathrm{i}} - O_{\mathrm{i}})^2 \tag{5}$$

$$\Delta W_{\mathrm{j}} = \eta(\frac{-\partial(E_{\mathrm{r}})}{\partial W_{\mathrm{j}}}) \tag{6}$$

$$W_{\mathrm{j}}(t+1) = W_{\mathrm{j}}(t) + \Delta W_{\mathrm{j}} \tag{7}$$

η represents the neural network learning rate. The role of this parameter is to control the amount of weight change at each stage [15].

In general, the value of the Mean Square Error (MSE) indicates how a neural network operates and is calculated by equation (8).

$$MSE = \frac{1}{n}\Sigma_{i=1}^{n}(T_{\mathrm{i}} - O_{\mathrm{i}})^2 \tag{8}$$

The typical BP algorithm adjusts the weights of the neural network in a Gradient Descent (GD) mode that means it minimizes the error between the actual output and the expected output for a specific input. A neural network with BP algorithm tries to converge to the global minimum point with the shortest path throughout the training process. This point is the maximum performance that the network can achieve depending on its size. The network learns quickly at the beginning of the learning process and the value of MSE decreases rapidly. Since as the learning process progresses, the learning speed of the neural network decreases, this type of network requires a high number of iterations to learn the data with a desired degree of accuracy [20].

According to what was discussed, it is obvious that the GD method is a solution with slow convergence rate [8] and there are two major types of problems in BP algorithm, i.e., its slow convergence rate and the possibility of being at a local minimum [22].

3 Using Fuzzy Logic for Increasing the Speed of Neural Network with BP Algorithm

People often think that the accuracy of the simulation and the speed of convergence of the neural network are directly proportional to the degree of complexity of its structure. The more complex the neural network topology and the more hidden layer nodes, the greater accuracy of the simulation and the less possibility of falling into the local minimum points. Therefore, in practical applications, people increase the number of hidden layer nodes indefinitely. They even use the layer nested overlap network to simulate and solve ordinary problems. But the results of many experiments performed by selecting some generic examples show that the accuracy of the simulation and speed of neural network convergence will not increase with augmentation of the degree of complexity. For example, the performance error of a BP algorithm simulation model with the different number of hidden layer nodes can be seen in Fig. 3. The number of output layer nodes is 1 and the number of input layer nodes is 2. The number of hidden layer nodes varies from 1 to 5. Fig. 3. Illustrates the convergence errors [19].

Fig. 3. Shows that when the number of hidden layer nodes is considered as an odd number, the simulation errors are larger and this is not convergence. In addition, in this case, the errors increase with augmenting the number of hidden layer nodes. But, when the number of hidden layer nodes is considered as an even number, the simulation errors are small and rapidly converging. Also, in such case, as the hidden layer nodes increase, the error decreases, and consequently the speed of convergence increases. This means that the simulation accuracy and neural network convergence speed are not always directly proportional to the number of hidden layer neurons or the degree of complexity of the neural network structure [19].

According to what mentioned earlier, due to the nature of the data, hidden layers with a lot of nodes may add a large amount of time to the training phase and make no significant reduction in MSE. Also, according to the fact that a multilayer neural network with more than one hidden layer is slow in the training phase and also will not have good results [4], increasing the number of hidden layers cannot be a good way to increase the neural network

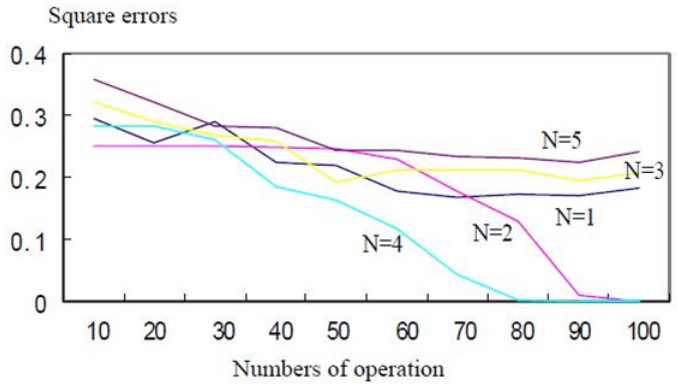

Figure 3: An example of the BP algorithm simulation with the different number of hidden layer nodes [19]

convergence speed. Therefore, we need another solution to increase the speed of convergence in the neural network with BP algorithm.

3.1 Fuzzy Logic Controller

Regarding the speed of convergence of the neural network with BP algorithm, not only the network structure is a significant factor, but also the learning rate and initial weights strongly affect the convergence speed and capability of the neural network [19]. Achieving a fuzzy logic control system to specify the learning rate and increase the speed of convergence in neural network with BP algorithm is the focus of this section. This contradicts with the normal BP neural network where the constant learning parameter is assumed. To adjust the learning rate based on the MSE error produced by the neural network, the fuzzy logic controller is employed. A fundamental architecture for such a monolith system is indicated in Fig. 4.

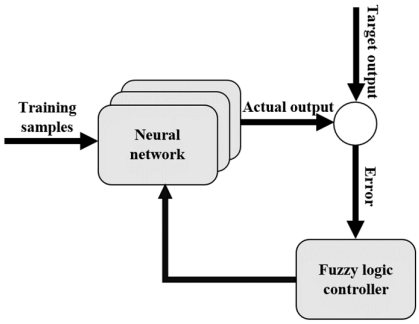

Figure 4: A fundamental architecture of a neural network controlled by the fuzzy logic controller [20]

Fuzzy logic control systems are preferred for two reasons, imprecise (ambiguous) linguistic descriptions and unlimited measurement accuracy. Using linguistic variables and fuzzy sets, there are no accuracy limits in a typical control system. In addition, using language variables, fuzzy control systems can include complex input and output processes as rules. Also, the functions of most control systems are founded on "expert knowledge" [20]. Moreover, in general, the fuzzy approach is explicit and systemic [21].

A fuzzy system, as shown in Fig. 5, has four main sections. The function of which is as follows:

First, the crisp inputs through the fuzzy membership functions are converted to fuzzy values. Then, the inference engine performs a logical operation on the fuzzy inputs using the if-then rules stored in the knowledge base and combines the multiple results into an interpolated result by grading the rules. The results are defuzzified again using membership functions for generating the crisp output.

In this paper, the task of the fuzzy logic controller is to automatically control the learning rate pursuant to the MSE level. Accordingly, the fuzzy controller input is error of each iteration of algorithm and output is the learning rate for the subsequent iteration of algorithm. The range of input and output must first be specified. According to most resources, the learning rate has a small positive value between 0 and 1. Although, it has been observed that this

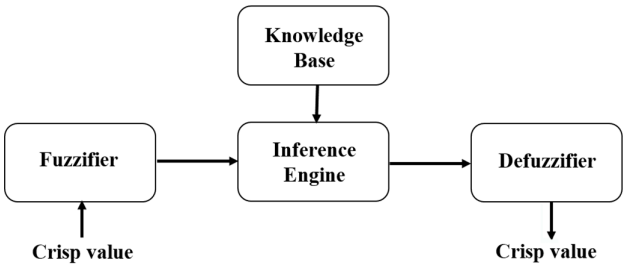

Figure 5: A generic design of a fuzzy logic controller [27] and [7]

parameter actually takes higher values in the GDA method steps in MATLAB software. However, in order to have the possibility of generalizing the results of this paper to any similar simulation with a different type of problem or data set and also to avoid any operational risks, the range of the learning rate parameter in this research is considered between 0 and 1. Regarding the error range, this value should be determined based on the type of problem and data set being tested. Here, the error range is intended between ± 0.1.

After determining the range for the input and output of the fuzzy controller, it is necessary to divide each range as follows and define the membership functions:

- Learning rate: zero, small, medium, large
- Error: large & medium & small negative, zero, small & medium & large positive

According to the theory that "if the error is large, it is necessary to increase the learning rate and if the error is small and approaching the optimal response, we are required to take smaller steps and decrease the learning rate", we adjust the fuzzy rules according to the Table 1 [11].

Table 1: Fuzzy rules for controlling the learning rate

Rule	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Error	Negative Large	Negative Medium	Negative Small	Zero	Positive Small	Positive Medium	Positive Large
Learning Rate	Large	Medium	Small	Zero	Small	Medium	Large

Fig. 6. To Fig. 8. Illustrate an example of fuzzy membership functions for error as input and learning rate as output as well as fuzzy rules.

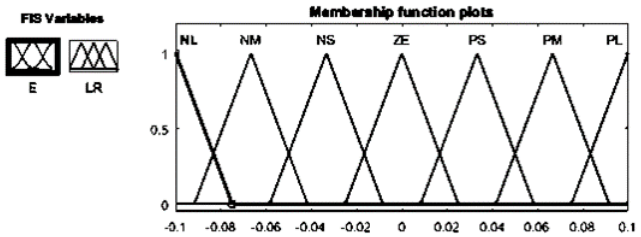

Figure 6: Membership functions for error

4 Experimental Results

4.1 Fuzzy Clustering

In implementation step of this paper, a dataset has been used in which important requirements for producing a series of effective data are identified and exist. Important scenarios of attack and injection method are also included. The investigation of the Simple Network Management Protocol (SNMP) for network anomaly detection is the systematic

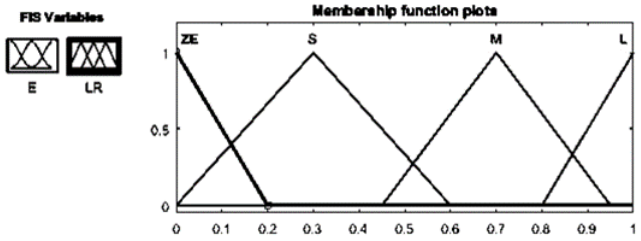

Figure 7: Membership functions for learning rate

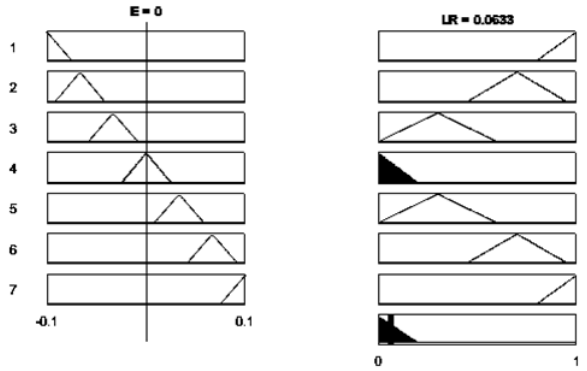

Figure 8: Fuzzy rules to control the learning rate based on error

approach to providing this dataset. This dataset has 4,998 samples each includes of 34 Management Information Base (MIB) variables, which are divided into 8 clusters [2].

FCM algorithm is used in MATLAB software to implement fuzzy clustering on this dataset. At first, all the samples are given to the software, each a combination of 34 various characteristics. Normalization process is made on the dataset to coordinate various features and different ranges. Then, clustering is carried out based on intended parameters. These parameters include overlap value of fuzzy clusters (m), maximum number of iterations to achieve minimum objective function and minimum improvement in objective function between two consecutive iterations. In this simulation, overlap value is set at 2.5, maximum number of iterations at 1,000 and minimum improvement in objective function between consecutive iterations as $1*10^{-5}$. Moreover, number of considered clusters as an input parameter is set at 8. Centers of clusters and membership value of each training sample in any of the clusters are calculated based on the determined parameters and are returned as output. Fig.9. Shows the diagram for clustering the dataset in question based on two features, namely the number of packets sent by a sub-layer to a higher sub-layer or layer which were not addressed to a multicast or broadcast address at this sub-layer and the number of packets sent by a sub-layer to a higher sub-layer or layer which were addressed to a multicast or broadcast address at this sub-layer.

As seen in Fig. 9, each sample with different membership values is allocated to all the 8 clusters. In case of the red cluster for instance, it is observed that some of its members have a higher degree of dependence and have taken the red color more than any other colors. In contrast, some other samples with a lower degree of dependence on the cluster have taken less amount of red color. This capability allows for the possibility of a fuzzy view which is near to human view to the set of network faults.

It should be noted that overlap value of fuzzy clusters in this simulation is set at 2.5 after several stages of clustering test on the dataset. This is because in case of a higher number, overlap value increases and some of the clusters coincide and one of them lose their members completely. By increasing this number in fact, the algorithm tends from fuzzy to non-fuzzy mode. Moreover, by decreasing the number, objective function increases and gets far from the most preferred possible state. Since the value of objective function is influenced by both the number of samples and the number of clusters and further to the fact that number of samples in the dataset in question is nearly 5,000 samples with 8 clusters, output value of objective function in this simulation is near to 500.

After completion of fuzzy clustering, we have the results of different states for combination of features; i.e., different samples in the dataset, as target and training samples in the dataset as input for training the BP neural network.

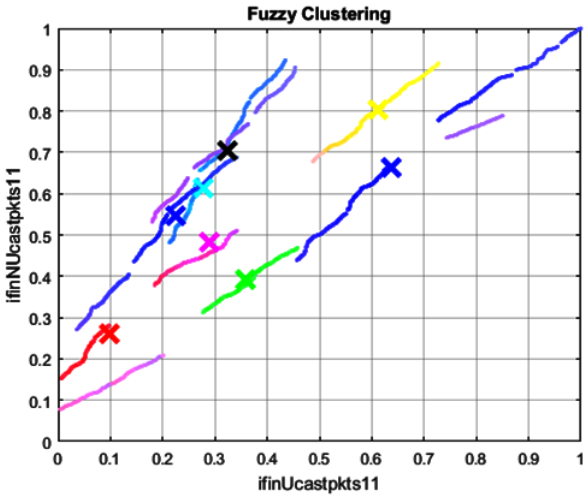

Figure 9: Dataset clustering diagram based on two of the input features

4.2 Updating the Learning Rate of BP Neural Network by Fuzzy Logic Controller

Upon completion of fuzzy clustering process, we have a table consisting of all samples and dependence level of each to any of the clusters indicating a state of network and its faults. This table will be used as the target of dataset for BP neural network. At this stage, we prepare a fuzzy controller with an error range of ±0.1 to control algorithm learning rate through Matlab fuzzy tool. Then, we put BP neural network in three training states and compare the results. We use traingd function for one time. In this function, learning rate throughout the entire learning process is a constant number which we take it as 0.1. In the next stage, we use traingda function which tries to improve the speed of this process through certain coefficients to decrease and/or increase the learning rate during learning process. Again, we take the learning rate as 0.1 at the beginning of the process and we take increase and decrease coefficients as 1.05 and 0.7, respectively. At the final stage, we use traingd function once again with an initial learning rate of 0.1, but we determine for the function to receive a new learning rate from the fuzzy controller in each iteration of learning loop. The results of all the three learning states can be seen in the diagram of Fig. 10.

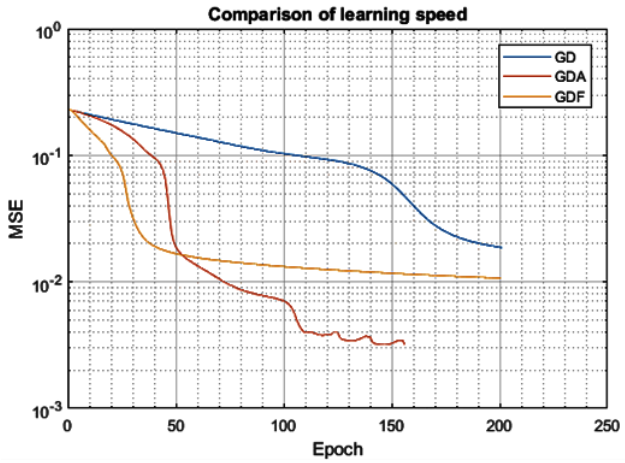

Figure 10: Comparison of learning speed of GD, GDA and GDF with single fuzzy controller in 200 epochs

As it is clearly seen in the diagram of Fig. 10, traingd function with the fuzzy controller to which we refer as GDF descends to a lower error level with a higher speed in the first 50 epochs; i.e., it acts better at the beginning of learning process as compared to GD function with a constant learning rate and GDA function with a variable learning rate based on the specified decreasing and increasing coefficients. GDF continues such decreasing trend up to the end of training process, but its convergence speed decreases gradually; i.e., as the number of iterations is increased and the range of error gets smaller, the learning rate considered for each epoch by the fuzzy controller has no significant change as compared to the past stage and this change does not include a large number. For this reason, GDF convergence

speed decreases gradually and although it acts better up to the end of learning process as compared to GD function with a constant learning rate, performance of GDA function is better than GDF method after epoch No. 50.

We repeat simulation for another time with similar conditions to re-improve the proposed GDF procedure. But this time, we consider several fuzzy controllers for various error ranges instead of just one fuzzy controller and we determine for the algorithm to receive its next iteration learning rate each time from the fuzzy controller relevant to the same error range by considering the error range. The results obtained from this simulation for all the three learning states can be seen in the diagram of Fig. 11. Also, the error values in the epochs 0, 50, 100, 150 and 200 for each of the 4 learning modes are included in Table 2.

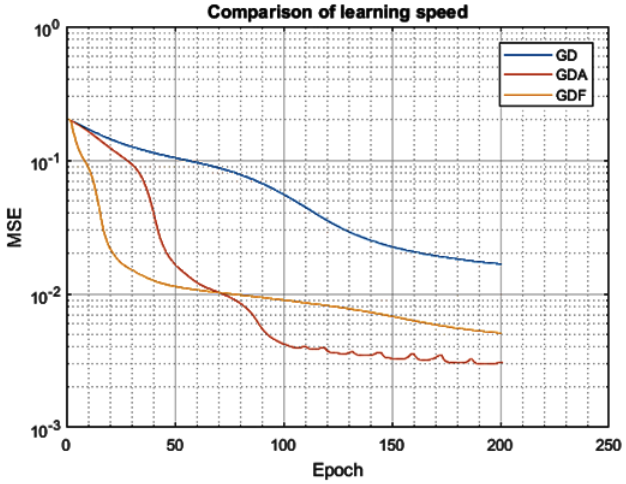

Figure 11: Comparison of learning speed of GD, GDA and GDF with multi fuzzy controller in 200 epochs

Table 2: The error values for the 4 learning modes

	Number of Epoch				
Learning Mode	**0**	**50**	**100**	**150**	**200**
GD	0.2599	0.1698	0.0798	0.0453	0.0366
GDA	0.2599	0.0358	0.0052	0.0035	0.0033
GDF (Single)	0.2599	0.0351	0.0217	0.0162	0.0136
GDF (multi)	0.2599	0.0207	0.0120	0.0086	0.0066

As seen in the diagram of Fig. 11, performance of GDF procedure improves upon an increase in the number of controllers and its convergence speed has a significant increase as compared to GD with a constant learning rate. Moreover, it has been better than the single controller state in competition with GDA function. GDA has once again taken larger steps towards learning the algorithm by making larger changes in the learning rate and this has caused a cross-sectional increase in performance error in some parts of the GDA diagram. However, changes of learning rate in GDF procedure are still smaller than GDA. Therefore, it has a strictly descending diagram which indicates the accuracy of general logic of this research.

5 Conclusion and Suggestions

According to above discussion, we managed to use fuzzy logic to improve the performance of BP neural network in automatic fault diagnosing in computer networks. At first, we used Fuzzy C-Means to provide fuzzy inductive results from the intended dataset to train BP neural network. Then and in order to increase the training speed of BP neural network, we used fuzzy controller to adjust learning rate and we achieved successful results.

Moreover, we may use genetic algorithm in future researches to improve the performance of both sections discussed in this paper. The statistical hypothesis testing illustrates that doing feature selection phase before the learning

process of the BP algorithm, increases the performance of this algorithm [10]. So genetic algorithm can choose the input features without any changes to decrease the input dimensions and to apply them in the network training process thus affecting both the accuracy of fuzzy clustering and the training speed of BP neural network. On the other hand and considering the point that in order to achieve higher precision from the fuzzy controller it is necessary to update and ameliorate the fuzzy structure [24], the genetic algorithm can update the parameters of fuzzy inference model in each iteration of BP algorithm by considering the feedback we receive from each iteration of BP algorithm. They include the parameters of membership functions, type of membership functions and performance range of membership functions. It is expected that in such case, we may see quite better feedback as compared to GDA procedure.

References

[1] F. Ahmad, Z. Ahmad, C.A. Kerrache, F. Kurugollu, A. Adnane, and E. Barka, *Blockchain in internet-of-things: Architecture, applications and research directions*, Int. Conf. Comput. Inf. Sci., 2019, 6 Pages.

[2] M. Al-Kasassbeh, G. Al-Naymat, and E. Al-Hawari, *Towards generating realistic SNMP-MIB dataset for network anomaly detection*, Int. J. Comput. Sci. Inf. Secur. **14** (2016), no. 9.

[3] A. Bagherinia, B. Minaei-Bidgoli, M. Hossinzadeh, and H. Parvin, *Elite fuzzy clustering ensemble based on clustering diversity and quality measures*, Appl. Intell.**49** (2019), 1724–1747.

[4] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, *Greedy layer-wise training of deep networks*, 19th Int. Conf. Neural Inf. Process. Syst., December, 2006, 8 Pages.

[5] J.C. Bezdek, R. Ehrlich, and W. Full, *FCM: The fuzzy c-means clustering algorithm*, Comput. Geosci. **10** (1984), no. 2-3, 191–203.

[6] J.C. Bezdek, R.J. Hathaway, M.J. Sabin, and W.T. Tucker, *Convergence theory for fuzzy c-means: Counterexamples and repairs*, IEEE Trans. Syst. Man Cybernet. **SMC-17** (1987), no. 5.

[7] M.C. Choy, D. Srinivasan, and R.L. Cheu, *Neural networks for continuous online learning and control*, IEEE Trans. Neural Networks **17** (2006), no. 6, 1511–1531.

[8] M. Georgiopoulos, C. Li, and T. Kocak, *Learning in the feed-forward random neural network: A critical review*, Perform. Eval. **68** (2011), no. 4, 361–384.

[9] N. Goyal, M. Dave, and A.K. Verma, *Fuzzy based clustering and aggregation technique for under water wireless sensor networks*, Int. Conf. Electron. Commun. Syst., 2014, 5 Pages.

[10] C-M. Hsu, *Forecasting stock/futures prices by using neural networks with feature selection*, 6th IEEE Joint Int. Inf. Technol. Artific. Intell. Conf., August 2011, 7 Pages.

[11] Q. Hu, and D.B. Hertz, *Fuzzy logic controlled neural network learning*, Inf. Sci. **2** (1994), no. 1, 15–33.

[12] J.P. Jesan, *The Neural Approach to Pattern Recognition*, ACM Digital Library, April 2004, `https://ubiquity.acm.org/article.cfm?id=985625`.

[13] T. Kaur, *Implementation of backpropagation algorithm: A neural network approach for pattern recognition*, Int. J. Engin. Res. Dev. **1** (2012), no. 5, 30–37.

[14] A.S. Khan, Z. Ahmad, J. Abdullah, and F. Ahmad, *A spectrogram image-based network anomaly detection system using deep convolutional neural network*, IEEE Access **9** (2021), 87079–87093.

[15] S.Y. Kung, and J.N. Hwang, *An algebraic projection analysis for optimal hidden units size and learning rates in back-propagation learning*, IEEE Int. Conf. Neural Networks, 1988, 8 Pages.

[16] C.Y. Lee, M.S. Wen, G.L. Zhou, and T.A. Le, *Application of ANN in induction-motor fault-detection system established with MRA and CFFS*, Mathematics **10** (2022), no. 13, 2250.

[17] M. Madhiarasan and S.N. Deepa, *A Novel Criterion to Select Hidden Neuron Numbers in Improved Back Propagation Networks for Wind Speed Forecasting*, Appl. Intell. **44** (2016), 878–893.

[18] K. Qader and M. Adda, *Network faults classification using FCM*, 17th Int. Conf. Distributed Comput. Commun. Networks (DCCN), 2013, 8 Pages.

[19] M. Qingwu, L. Chengbin, C. Hu, and L. Ti, *Improved BP network algorithm based on fuzzy logic and application in geophysics*, Eur. Symp. Intell. Techniq., September, 2000, 8 Pages.

[20] H. Rashidy Kanan, and M. Yousefi Azar Khanian, *Reduction of neural network training time using an adaptive fuzzy approach in real time applications*, Int. J. Inf. Electron. Engin. **2** (2012), no. 3.

[21] P.D. Raval, and A.S. Pandya, *Improved fault classification in series compensated EHV transmission line using wavelet transform and artificial neural network*, IEEE 1st Int. Conf. Power Electron. Intell. Control Energy Syst., July, 2016, 5 Pages.

[22] D. Srinivasan, X. Jin, and R.L. Cheu, *Adaptive neural network models for automatic incident detection on freeways*, Neurocomputing **64** (2005), 473–496.

[23] S.N. Sivanandam, S. Sumathi, and S.N. Deepa, *Introduction to Neural Networks using Matlab 6.0*, 1st ed., Tata McGraw Hill, India, 2006.

[24] C. Van Kien, H.P.H. Anh, and N.N. Son, *Adaptive inverse multilayer fuzzy control for uncertain nonlinear system optimizing with differential devolution algorithm*, Appl. Intell. **51** (2021), 527–548.

[25] R. Wan, N. Xiong, Q. Hu, H.Wang, and J. Shang, *Similarity-aware data aggregation using fuzzy c-means approach for wireless sensor networks*, EURASIP J. Wireless Commun. Network. **2019** (2019), 1–11.

[26] Q. Wang, *Computer network fault diagnosis based on neural network*, Int. J. Future Gener. Commun. Network. **8** (2015), no. 5, 39–50.

[27] J. Zhang, *Modelling and optimal control of batch processes using recurrent neuro-fuzzy networks*, IEEE Trans. Fuzzy Syst. **13** (2005), no. 4, 417–427.