JEAS

JOURNAL OF ENGINEERING AND APPLIED SCIENCES

Journal of Engineering and Applied Sciences 12 (2017), 94-102

Performance Metrics of Various Topologies of a Feed Forward Error-Back Propagation Neural Network

M.S. Osigbemeh[1*], C.C. Okezie[2] and H.C. Inyiama[3]
[1]Dept of Electrical and Electronics, Federal University Ndufu-Alike Ikwo.
[2]Dept of Electronic and Computer Engineering, Nnamdi Azikiwe University. Awka.
[3]Dept of Electronic and Computer Engineering, Nnamdi Azikiwe University. Awka.
[*]Corresponding Author's E-mail: eosigbemeh@yahoo.com

Abstract
The use of artificial neural network in processing of information has continued to become a robust tool of choice for researchers especially for the modeling of both real-valued and vector-valued functions over continuous and discrete-valued attributes with ability to absolve noise in the training and validation data. This paper demonstrates some of the findings in the implementation of a fully connected feedforward error-back propagation artificial neural network on a range of pre-normalized inputs, with their corresponding output parameters and the deductions obtained from the various iterative tests performed with varying learning rate, η. A significant speed in network convergence and an appreciable error tolerance was achieved when η was set to 0.4. The neural network's precision and F-score based on computing the confusion matrix of all iteration sessions was also used to analyzed the performance metrics of the investigated artificial neural network.

Keywords: Neural Network Topology, Gradient Descent Optimization, Supervised Learning, Backpropagation, Numerical optimization.

1. Introduction

As database systems continue to become flawless and robust in their designs creating features such as improved security, increased capacity with efficient data handling and logging, it has become necessary to implement data mining rules and principles capable of making sense from these databases, this is coming on the dawn of the availability of cheap and efficient storage devices with capacities of several hundreds of terabytes in consumer electronics. These devices have made accumulation of data in the form of audio and video streams, metadata, pictures, different file formats and types possible with high fidelity. The present future will require robust solutions that will help in the mining or making sense out of these avalanches of stored resource-rich data mines. The connectionist approach to data mining provided by artificial neural network, ANN has made it very possible to link outputs to inputted parameters of a modeled system without having an understanding of the connection or mathematical relationship of such systems under investigation. This is achieved sometimes with lower classification error rate when compared with other methods such as decision trees when used for data classification. The mathematical relationship of systems especially nonlinear systems have continued to be a clog in engineering training and practice for decades making would-be students of engineering to jettison the field for other less demanding areas or disciplines. Though the mathematical analysis of systems' behavior is necessary even for their computational merits at least in their application to machine learning and other areas, the data processing ability of ANN have shown its ability to yield promising results in any area of implementation provided that the controlling conditions and laws of convention are adhered to by the user.
The major problem of obtaining the large training sets and also validation sets for inputs into the network for learning purpose have already been surmounted by the availability of huge storage devices as already pointed out above. The other issue of making sense of these data sources is less an herculean task as more information and experiments continue to be made available to the data mining community from different researcher's effort. Another aspect of the evolution of ANN computing is the adaptation of network weights and the topological structure which

are mainly dependent on heuristics or trial and error (Basheer & Hajmeer, 2000), (Bhadeshia, 1999), (Callan, 1999) and (Galushkin, 2007). It is worthy to note that ANNs do not solve all engineering problems as other machine learning techniques such as particle swarm optimization, genetic algorithms, fuzzy logic, etc. continue to emerge and sometimes provide solutions that are faster, may be more efficient and more reliable than those produced by ANN.

ANNs have been considered a 'black boxed' machine learning technique since it does not allow humans to peek into its processing mechanisms when implementing the back propagation algorithm, BPA (Mitchell, 1997), (Larose, 2005, 2006), (Rudin & Wagstaff, 2013) and (Rumelhart & McClelland, 1986), a feat many consider a disadvantage especially when deployed for highly sensitive data processing in the field of data mining which is increasingly been used to solve many challenges devoid of governing mathematical equations (Zhou, 2015). Authors like Lu et al. (1996), in an attempt to demystify the black box model of neural networks, had shown that concise symbolic rules similar to decision trees' can be extracted from neural networks. They argue that ANNs "should have their position in data mining because of its merits such as low classification error rates and robustness to noise." They however lamented the huge training time involved in converging to an optimal solution stating that the time of rules extraction even by their effort was still longer than that of decision trees. Furthermore, attempts to reduce computing time or complexities in processing have necessitated several modifications to the gradient descent algorithm in works like the Levenberg-Marquardt Algorithm, LMA (Hagan, et al. 2013), Hopfield Networks (Hopfield, 1984, Van Rooji et al., 1996); Adaptive Resonance Theory, ART (Carpenter and Grossberg, 1988); Radial Basis Function, RBF (Haykin, 1994); Kohenen Networks (Kohenen, 1989), Recurrent Networks (Pham, 1994), etc. each having its unique features.

The fastest known implementation of the modified gradient descent methodology is the LMA which can train neural networks 100 times faster than normal BPA (Badri, 2010), (Demuth & Beale, 2002) and (Hagan & Menhaj, 1999). The LMA which is used in MATLAB for processing neural networks can achieve convergence even in one or few iteration(s) depending on fed inputs. The LMA is a numerical optimization method for multilayer network training that operates by computing the Jacobian matrix from the derivatives of the generated errors, instead of the derivatives of the squared errors with respect to the weights and biases of the network as in standard BPA. Thus, it uses a variation of the backpropagation algorithm during operation (Hagan, et al. 2013) to allow very quick convergence with respect to training outputs. The dependence on the LMA and other modifications of the BPA though solves certain aspects of implementing neural networks, further creates the problem of making neural networks deeper in the "black box" concept which is a disadvantage for sensitive processing of data as it may impede their application to solving certain real and complex tasks. For instance, the use of ANN was employed by (Topping et al. 1998) for an efficient partitioning of adaptive unstructured finite element meshes using mean field annealing approach over hill-climbing simulated annealing and used for machine condition monitoring by (Gao, 2003).

Though the dependence or implementations of ANN continue to find its way into undergraduate curriculum with its attendant myth of the black box concept more can still be achieved in x-raying the black box to show the workings of the underlying concepts in the execution of the BPA. We attempt to support this direction in this work by publishing our findings in investigating several topologies using the standard BPA.

2.0 Material and methods

The standard BPA consists of computing the derivatives of the squared errors generated with respect to the weights and biases of the network during the first iteration in a feed forward scheme. The BPA must be presented with a pair of pre normalized initial inputs representing the physical parameters of concern to be analyzed by the network and the expected network's output solution based on the initial inputs. This computed error which is obtained by comparing the actual output with the expected output to obtain a difference (the error) is then used to adjust all generated weights and bias (during the first network sweep or iteration) in a backward or feedback manner. The error produced in the second network sweep is also used to obtain the direction in which the weights will be adjusted towards the expected output or solution. This is repeated for a specific number of iterations or epochs until a trend is noticed or a convergence is sought which then complete a network session. When the designed ANN is trained with similar data and corresponding expected solution the ANN is expected to converge to a specific solution similar to the learned session in around the same number of iterations used in its training since the weights (and bias) were randomly generated during the first forward sweep of the network. At this converged point, the network is said to be generalizable to new data since it is able to converge to solution in almost the same way it was trained. The network may disadvantageously memorize data which may result when trained with very similar input data and two few hidden nodes or hidden layers. A whole lot of issues crop up in ascertaining the memorizability of ANN when presented with a range of input data.

In the general training and validation of ANN, a huge number of training data sets and another set of validation data (about 20% of training data) could be used for a successful designing process. The numerous training sessions of the network will enable the ANN to learn the similarities embedded in the different inputted data at each presentation to the network. This is similar to the biological identification process in organisms such as animals (or humans) with the ability to identify something or someone from a continuously presented scenario in real-time and over a period of time. The ANN (as least by its proponents) attempts to mimic this biological process of identification by creating a 'trained set of weights' obtained from several training sessions which is then used to quickly obtain a solution in a 'single' iteration with a new set of input sequence. The several training sessions is similar to the time it takes for learning or becoming familiar with something or someone in biological identification. The trained set of weights corresponds with the complex biological process of identification by organisms at a 'single' glance. The sum of squared error, SSE approach to solving the standard BPA still stands as a more simplified way for arriving at a convergence solution. The SSE method is also advantageous as it provides an easily verifiable procedure for analyzing the implemented neural network and provides an option for easily adjusting the learning rate, η. The learning rate $0 < \eta < 1$ is an arbitrary constant chosen to help get the neural network weights towards its global minimum for SSE (Wikipedia. 2015) and must be chosen in such a way that it is neither too small or two large. Two large will lead to very large oscillations that makes converging to the global minimum impossible while too low will lead to an extremely long time for network convergence which is not a bearable option to the former.
The deployed ANN for this research, which was based on the SSE concept discussed above was developed and implemented by the researchers using the Visual Basic Studio programming language. The software which is available in executable standalone file can run in any Microsoft Windows OS and features graphical displays or plots, options for adjusting randomly-generated weight's ranges; learning rates, η and real-time update of weights on the graphical user interface, GUI as shown if Figure 1 (a).

3.0 Results and Discussions

In the analysis of various topologies or architectures for implementing the neural network with respect to the number of hidden nodes in a hidden layer, the following results were obtained. Amongst the investigated ANNs includes the one hidden node, two hidden nodes, three hidden nodes and five hidden nodes' ANN responses to pre-defined inputs with the input values ranging between 0.1 and 0.9. The inputs represent the presented physical parameter of interest to be analyzed by the ANN and could range from pre-fuzzified data or pre-processed data, etc. In this research, fuzzified diagnostic data from fuzzy processing of patient's reported symptoms captured by the SOSIC Expert System for Contagious Disease Detection, SESCDT (Osigbemeh et al. 2014) have been used as inputs to the designed ANN. The various ANNs were fed with the same inputs in other to compare their convergence characteristics, sensitivity, execution time and generalization over training and validation data.
The network's precision was computed from the following formulation presented in Wikipedia. (2016), Iwasokun et al, (2015), and (Powers, 2011).

$$PPV = {TP}/{TP + FP} \tag{1.1}$$

and the Sensistivity, Recall or True Positive Rate (TPR) was computed from

$$TPR = {TP}/{TP + TN} \tag{1.2}$$

and the F-Score which is the harmonic mean of precision and sensitivity was computed from

$$\text{F - Score} = {2TP}/{2TP + FP + FN} \tag{1.3}$$

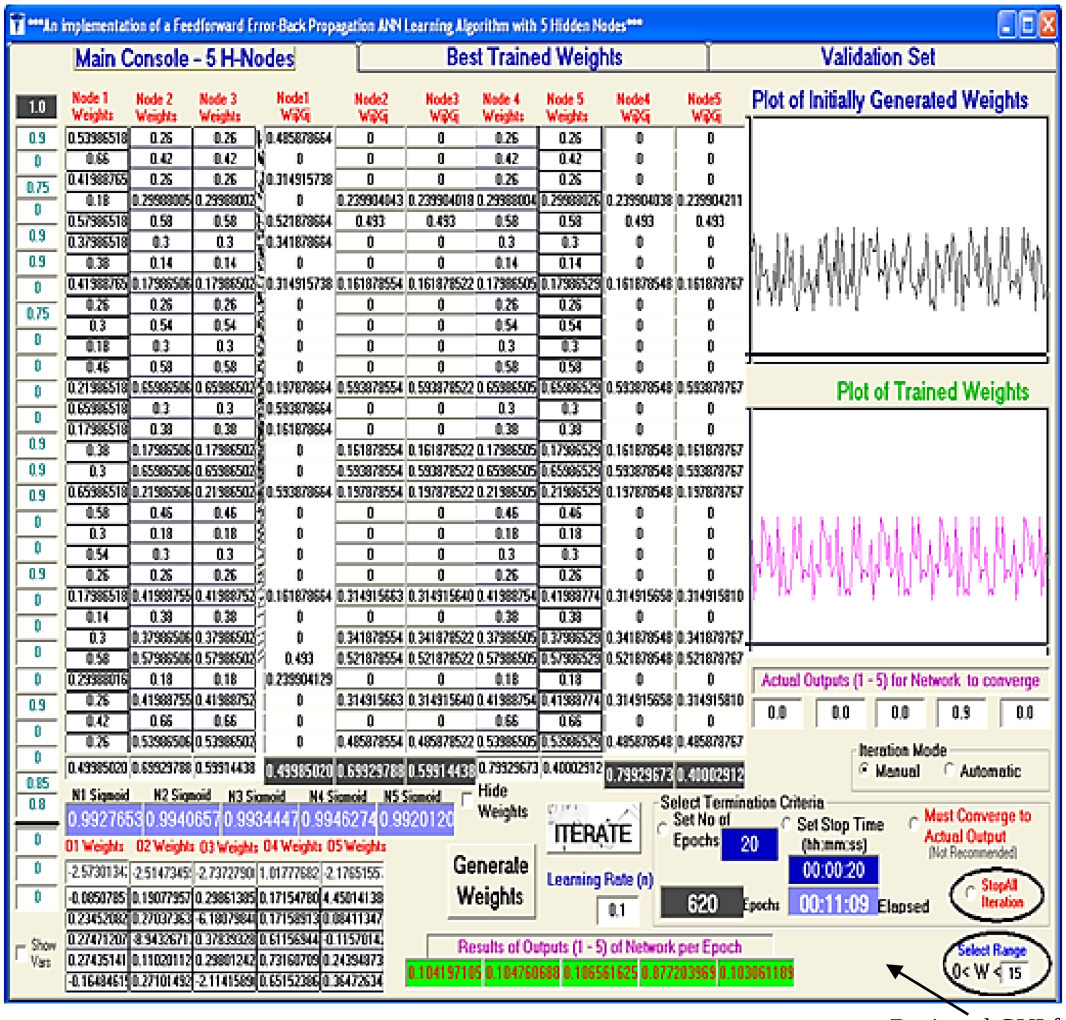
Where FP = false positive, TN = true positive, FN = false negative and TP = true positive.
The results which are tabulated below showed that network precision on fed data showed significant precision at over 98% with learning rate, η = 0.1 and over 80% when η increased to 0.9 with corresponding decrease of the F-statistics. At η = 0.3 and 0.4, precision was maintained at over 90% which showed that the network performance was still at optimal based on the fed data to the neural network.

3.1 Data Preparation

A total of 1185 records of the SESCDT processed diagnostic data was used as inputs to the various sessions of each topology of the ANN. Out of the total inputs 15% were set aside for validation while another 10% was set aside for testing of the ANN performance. In each session, the ANN response and the best convergence weights for each topology was obtained and collated. The ANN was also presented with the expected result to converge to while a notice was made of the number of successive iterations the particular ANN had executed in arriving at a convergence solution as shown in Figure 1 (a). A table of the effect of variation of the learning rate, η from 0.1 to the maximum of 0.9 for all considered nodal topologies and their corresponding convergence epochs was also investigated below.

3.1.1 Performance Metrics for Five Hidden Nodes

In the five-node topology, a benchmark epoch of 620th epoch was realized in the training sessions for convergence of inputted pre-fuzzified contagious-diseases-diagnosed symptoms with a corresponding expected output result for training the ANN. It was observed that execution time for up to the 620th benchmark iteration of the ANN was nominally at around 00:11:00 and a high sensitivity was obtained by the five-node neural network which was trained at the minimum learning rate of η = 0.1. However, generalization of the five-node ANN when presented with the remaining 25% input (15% validation and 10% testing) data was observed. It was also observed that the result of unconnected input data fed to the ANN processing was unstable leading to convergence before or generation of significant errors around the 620th epoch proving that the ANN had begun memorizing the input data samples.



1.0

0.9, 0, 0.75, 0, 0.9, 0.9, 0, 0.75, 0, 0, 0, 0, 0.9, 0.9, 0.9, 0, 0, 0.9, 0, 0, 0, 0, 0.9, 0, 0, 0.85, 0.8

0, 0, 0

Node 1 Weights	Node 2 Weights	Node 3 Weights	Node1 WjXj	Node2 WjXj	Node3 WjXj	Node 4 Weights	Node 5 Weights	Node4 WjXj	Node5 WjXj
0.53986518	0.26	0.26	0.485878664	0	0	0.26	0.26	0	0
0.66	0.42	0.42	0	0	0	0.42	0.42	0	0
0.41988765	0.26	0.26	0.314915738	0	0	0.26	0.26	0	0
0.18	0.29988005	0.29988002	0	0.239904043	0.239904018	0.29988004	0.29988026	0.239904038	0.239904211
0.57986518	0.58	0.58	0.521878664	0.493	0.493	0.58	0.58	0.493	0.493
0.37986518	0.3	0.3	0.341878664	0	0	0.3	0.3	0	0
0.38	0.14	0.14	0	0	0	0.14	0.14	0	0
0.41988765	0.17986506	0.17986502	0.314915738	0.161878554	0.161878522	0.17986505	0.17986529	0.161878548	0.161878767
0.26	0.26	0.26	0	0	0	0.26	0.26	0	0
0.3	0.54	0.54	0	0	0	0.54	0.54	0	0
0.18	0.3	0.3	0	0	0	0.3	0.3	0	0
0.46	0.58	0.58	0	0	0	0.58	0.58	0	0
0.21986518	0.65986506	0.65986502	0.197878664	0.593878554	0.593878522	0.65986505	0.65986529	0.593878548	0.593878767
0.65986518	0.3	0.3	0.593878664	0	0	0.3	0.3	0	0
0.17986518	0.38	0.38	0.161878664	0	0	0.38	0.38	0	0
0.38	0.17986506	0.17986502	0	0.161878554	0.161878522	0.17986505	0.17986529	0.161878548	0.161878767
0.3	0.65986506	0.65986502	0	0.593878554	0.593878522	0.65986505	0.65986529	0.593878548	0.593878767
0.65986518	0.21986506	0.21986502	0.593878664	0.197878554	0.197878522	0.21986505	0.21986529	0.197878548	0.197878767
0.58	0.46	0.46	0	0	0	0.46	0.46	0	0
0.3	0.18	0.18	0	0	0	0.18	0.18	0	0
0.54	0.3	0.3	0	0	0	0.3	0.3	0	0
0.26	0.26	0.26	0	0	0	0.26	0.26	0	0
0.17986518	0.41988755	0.41988752	0.161878664	0.314915663	0.314915640	0.41988754	0.41988774	0.314915658	0.314915810
0.14	0.38	0.38	0	0	0	0.38	0.38	0	0
0.3	0.37986506	0.37986502	0	0.341878554	0.341878522	0.37986505	0.37986529	0.341878548	0.341878767
0.58	0.57986506	0.57986502	0.493	0.521878554	0.521878522	0.57986505	0.57986529	0.521878548	0.521878767
0.29988016	0.18	0.18	0.239904129	0	0	0.18	0.18	0	0
0.26	0.41988755	0.41988752	0	0.314915663	0.314915640	0.41988754	0.41988774	0.314915658	0.314915810
0.42	0.66	0.66	0	0	0	0.66	0.66	0	0
0.26	0.53986506	0.53986502	0	0.485878554	0.485878522	0.53986505	0.53986529	0.485878548	0.485878767
0.49985020	0.69929788	0.59914438	0.49985020	0.69929788	0.59914438	0.79929673	0.40002912	0.79929673	0.40002912

N1 Sigmoid	N2 Sigmoid	N3 Sigmoid	N4 Sigmoid	N5 Sigmoid
0.9927653	0.9940657	0.9934447	0.9946274	0.9920120

O1 Weights	O2 Weights	O3 Weights	O4 Weights	O5 Weights
-2.5730134	-2.5147345	-2.7372790	1.01777682	-2.1765155
-0.0850785	0.19077957	0.29861385	0.17154780	4.45014138
0.23452082	0.27037363	-6.1807984	0.17158913	0.08411347
0.27471207	-8.9432671	0.37839328	0.61156944	-0.1157014
0.27435141	0.11020112	0.29801242	0.73160709	0.24394873
-0.1648461	0.27101492	-2.1141589	0.65152386	0.36472634

Learning Rate (η) 0.1

Actual Outputs (1 - 5) for Network to converge

0.0	0.0	0.0	0.9	0.0

Iteration Mode [✓ Manual] Automatic

Select Termination Criteria
Set No of Epochs 20
Set Stop Time (hh:mm:ss) 00:00:20
Must Converge to Actual Output (Not Recommended)
620 Epochs 00:11:09 Elapsed

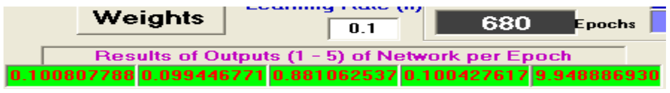
Results of Outputs (1 - 5) of Network per Epoch

0.104197105	0.104760688	0.106561625	0.877203969	0.103061189

Select Range 0< W < 15

(a)

Designed GUI for ANN Simulation

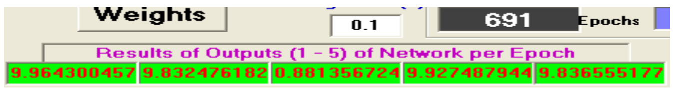


Weights 0.1 680 Epochs

Results of Outputs (1 - 5) of Network per Epoch

0.100807788	0.099446771	0.881062537	0.100427617	9.948886930

(b)



Weights 0.1 691 Epochs

Results of Outputs (1 - 5) of Network per Epoch

9.964300457	9.832476182	0.881356724	9.927487944	9.836555177

(c)

Figure 1: Screenshots of the results of iteration of a five-node hidden layer ANN at (a) 620th convergence Epoch (b) 680th error introduction Epoch and (c) 691st significant error Epoch

3.1.2 Performance Metrics for Three Hidden Nodes

In the performance metrics for a three-node hidden layer implementation of the feedforward error-back propagation learning ANN with pre-fuzzified inputs, convergence was already occurring at about five minutes into the session and around 306 epoch at the minimum learning rate of η = 0.1. Adequate convergence was attained around the 620th epoch. Further fine-tuning resulted in significant errors at around 663rd epoch. An appreciable sensitivity was observed when its results was compared with the five-node (above) session's results since the introduction of errors occurred at around the 663rd epoch while session time was at an average of 00:10:36. Performance of the three-node ANN on validation data showed good generalization and the result when fed with completely unconnected data showed significant errors at the 620th epoch.

3.1.3 Performance Metrics for Two Hidden Nodes

Adequate convergence was attained at the experimentally determined benchmark for the nature of the inputted diagnostic data at 620th epoch for the performance metrics for a two-node hidden layer of a fully connected feedforward error-back propagation learning ANN at the minimum learning rate, $\eta = 0.1$. Further fine-tuning resulted in significant errors at around 680th epoch. An insignificant sensitivity was observed when this result was compared with the previous three-node and five-node hidden layer architecture's result.

Further iterations beyond the 620th epoch produced more refinement towards the output data but eventually resulted in introduction of significant errors. These errors just as noticed above continue to increase within the outputs with further iterations.

3.1.4 Performance Metrics for One Hidden Node

Again, convergence was attained around the 620th epoch during training and beyond this benchmark; further fine-tuning resulted in significant errors at around the 741st epoch. These errors continue to propagate within the outputted actual-outputs with subsequent iterations thus making that ANN session null and void. The insensitivity of the one node hidden layer was noticed when the ANN continued to produce desired output far beyond the 620th epoch and becoming unstable in the 741st epoch thus indicating that utilizing the one node architecture may result in insensitivity to new data.

3.1.1.1 Summary of Performance Metrics of Considered ANNs

The adaptive tendencies to a desired output of the investigated ANN topologies with respect to certain input range have shown that the response of the five-node hidden layer ANN to new inputs suggests memorization of the network at the 620th benchmark epoch indicating that further increase in the number of hidden layers will continue to lead to data memorization. The three hidden node layer architecture had demostrated better generalization with less sensitvity when compared with the five-node. The one-node hidden layer topology though possesses good generalizability however demonstrated insensitive to a range of inputs that are similar to the training inputs, thus causing significant under fitting of intended data range. A significant improvement in sensitivity was obtained by the two-node ANN when compared with the one-node architecture during experiment. Thus the three-node ANN was chosen for further analysis according to (Suzuki et al. 2005), Hagan et al. (2013) and (Hassoun, 1995).
The learning rate, η has been left at the lowest value of 0.1 in all the scrutinized ANN topologies so as not to overstep the local minimum and to avoid large oscillations. This has made the number of iterations to become large at a bench mark of 620 epochs thus providing the best opportunity of the ANN to learn and the lowest opportunity to memorize. Table 1 shows a summary of the above performance metrics on the ANNs ability for generalization and sensitivity analysis with the one-node hidden layer ANN having the poorest sensitivity and the five-node hidden layer having the highest memorization possibility.

Table 1: Performance Metrics of various ANN topologies showing Sensitivity and Generalization on data.

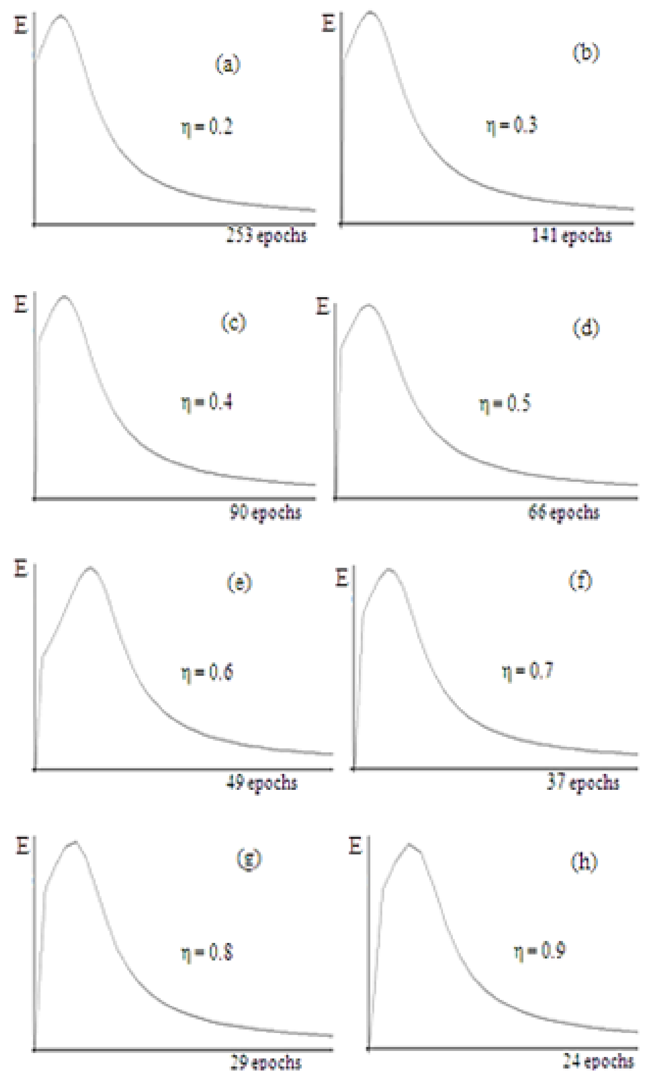
Hidden-Layer Topology	Average Time(s)	Sensitivity	Generalization	Memorization
One-Node ANN	10:23	Poor	Very Good	Very low
Two-Node ANN	10:36	Fair	Good	Low
Three-Node ANN	10:40	Good	Good	Normal
Five-Node ANN	10:58	Good	Poor	Highest

3.2 Effect of Increasing Learning Rate, η on Performance Metrics

The three-node hidden layer ANN was used in analyzing the effect of network performance on increasing learning rate by the network. The learning rate, η (analyzed above) which was initially chosen to be $\eta = 0.1$ to allow for very minimal oscillations during gradient descent optimization of the ANN, was varied by increasing η to $\eta = 0.2$, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8 and 0.9 respectively. In each increase of η, the performance of the ANN was observed and the benchmark iteration for that value of η was used in stopping the iterations just when the network's prediction accuracy is about to deteriorate or become error bound. A summary of the variations in learning rates for the training and validation data by the neural network is shown Table 2.

Table 2: Performance Metrics of variation of Learning rate, η with validation data

η	No of Epochs/Iterations	Time
0.1	620	10.36
0.2	253	04:18
0.3	141	02:26
0.4	90	01:35
0.5	66	01:11
0.6	49	00:52
0.7	37	00:40
0.8	29	00:28
0.9	24	00:24

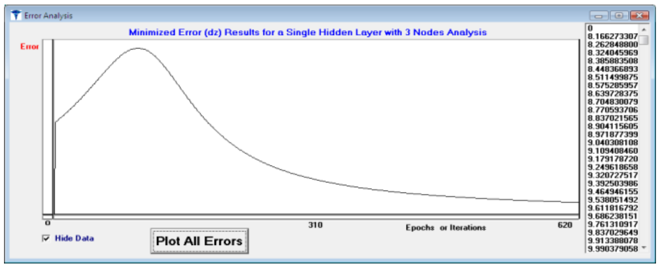


E represents the minimized error of the ANN session's iterations.

Figure 2: Screenshots of the symmetry of the graphs generated during increasing learning rate, η at (a) η = 0.2, (b) η = 0.3, (c) η = 0.4, (d) η = 0.5, (e) η = 0.6, (f) η = 0.7, (g) η = 0.8 and (h) η = 0.9 on input data.

Figure 2 and Figure 3 shows the corresponding graphical illustrations of the various increases in the values of η with a smooth symmetry noticed at Figure 3 and at Figure 2 (a), (b), and (c). However, as depicted, a coarse symmetry was also observed during iteration sessions as η increases from 0.4 through 0.9 as seen from (d) through (h).

These coarse symmetries are not desirous of the attributes of an efficient ANN that tries to improve speed by adjustment of learning rate as stated above and hence the 0.3 and 0.4 analysis for the values of learning rate was upheld in subsequent experiments. The value of 0.3 allowed for more accurate analysis while 0.4 yielded fastest and still tolerable analysis.


Error Analysis
Minimized Error (dz) Results for a Single Hidden Layer with 3 Nodes Analysis
Error
0
310
Epochs or Iterations
620
Hide Data
Plot All Errors

```
0
8.166273307
8.262848800
8.324045969
8.385883508
8.448366893
8.511499875
8.575285957
8.639728375
8.704830079
8.770593706
8.837021565
8.904115605
8.971877399
9.040308108
9.109408460
9.179178720
9.249618658
9.320727517
9.392503986
9.464946155
9.538051492
9.611816792
9.686238151
9.761310917
9.837029649
9.913388078
9.990379058
```

Figure 3: Screenshot of the symmetry of the graph generated during learning rate at η = 0.1 on inputted data.

Also in Table 2, the values of learning rate of η = 0.3 and 0.4 shows considerable but still tolerable deviations from the optimal trend in the number of epochs and total time spent by the ANN in executing the various iteration sessions. Table 3 reveals the results of computing the precision and F-statistics of the learning rates of η = 0.1 through 0.9 for a dataset of 1185 pre-fuzzified and normalized validation data. Again at η = 0.3 and 0.4, the precision was still upheld by the neural network at 0.982278 and 0.925738 respectively. Also at these learning rates, the F-statistics returned appreciable high values of 0.991060 for η = 0.3 and 0.961437 for η = 0.4. The values for precision and the F-statistics shows that at these learning rates, appreciable convergence could be sought for and achieved when the ANN is presented with similar input-output data pairs. Also the poor values of precision and F-statistics with further increase in the value of the learning rates can be found on Table 3 with the poorest values obtained when η = 0.9 with 0.804219 and 0.891487 for precision and F-statistics respectively.

Table 3: Results of processing of Precision and F-Score

SN	η	Total Records or Symptoms	TP	FP	TN	Precision	Recall	F-Score
1	0.1	1185	1169	16	0	0.986498	1	0.993203
2	0.2	1185	1167	18	0	0.984810	1	0.992347
3	0.3	1185	1164	21	0	0.982278	1	0.991060
4	0.4	1185	1097	88	0	0.925738	1	0.961437
5	0.5	1185	1060	125	0	0.894515	1	0.944321
6	0.6	1185	1034	151	0	0.872574	1	0.931951
7	0.7	1185	1016	169	0	0.857384	1	0.923217
8	0.8	1185	987	198	0	0.832911	1	0.908840
9	0.9	1185	953	232	0	0.804219	1	0.891487

4.0. Conclusion

The dependence on artificial neural networks for processing of a system's input-output pairs for efficient data mining capabilities have been investigated by this research. The fuzzy processing of patient's reported symptoms captured by the SOSIC Expert System for Contagious Disease Detection, SESCDT have provided the input data for validation purposes. The F-Score analysis and precision on the validation data was computed to appraise the designed neural network for which data from the SESCDT simulation software had been presented. The analysis showed over 95% precision rate which was expected as inputted data to the neural network was largely pre-fuzzified data consistent with contagious diseases infected patients. Also, the network's performance for various values of learning rate showed that the speed of network performance was enhanced when learning rate, η was at 0.3 and 0.4 while the enhanced convergence speed resulted to significant errors in just few iterations as learning rate increased from the duo towards 0.9. This was as a result of the neural network deteriorating in performance as it undergoes large oscillations in other to quickly get a solution and thus not been able to recognize the local minima of the solved problem.

5.0 Recommendation

Based on this research, data that can be pre-normalized, compressed or represented by the specified range of inputs acceptable by the developed ANN can be analyzed using any of the investigated topologies provided that the optimal convergence is attained during training sessions. Currently, active research is ongoing for implementing the results of this research on a case of pattern recognition and deep learning.

References

Basheer, I.A., & Hajmeer, M., 2000. Artificial Neural Network: Fundamentals, Computing, Design and Application. *Elsevier Journal of Microbiological Methods*, 43,1, 3 – 31.

Bhadeshia, H.K.D.H., 1999. Neural Networks in Materials Science. *ISIJ International*. 39,10, 966 – 979.

Callan, R., 1999. *The Essence of Neural Networks: Essence of Computing*, 1 – 57. Great Britain, Hertfordshire, HP2 7EZ.Prentice Hall Europe.

Carpenter, G.A., & Grossberg, S., 1988. *The ART of Adaptive Pattern Recognition by a Self-organizing Neural Network.* Computer March, 77 – 88.

Galushkin, A.I., 2007. *Neural Networks Theory*, 93 – 155. Germany: Springer-Verlag Berlin Heidelberg.

Gao, X.R., 2003. *Neural Networks for Machine Condition Monitoring and Fault Diagnosis*, 167 – 188. Department of Mechanical and Industrial Engineering, University of Massachusetts, USA. Retrieved on 8th July, 2014 from http://crema.di.unimi.it/fscotti/nn/8-gao-formatted.pdf.

Hagan, M.T., & Menhaj, M., 1999. Training Feed-Forward Networks with the Marquardt Algorithm. *IEEE Transactions on Neural Networks,* 5,6, 989 – 993.

Hagan, M.T, Demuth, H.B., & Beale, M.H., 1996. *Neural Network Design*, PWS Publishing Company, USA.

Hagan, M.T, Demuth, H.B, Beale, M.H., & DeJesus, O., 2013. *Neural Network Design*, 2nd Edition.

Hagan, M.T, Demuth, H.B., & DeJesus, O., 2002. An Introduction to the Use of Neural Networks in Control Systems. *International Journal of Robust and Nonlinear Control,* 12, 959 – 985. DOI: 10.1002/rnc.727.

Hassoun, M.H., 1995. *Fundamentals of Artificial Neural Networks.* MIT Press, Cambridge, MA.

Haykin, S., 1994. *Neural Networks: A Comprehensive Foundation.* Maccillan, New York.

Hopfield, J.J., 1984. Neurons with Graded Response have Collective Computational Properties like Those of Two-state Neurons. *Proceedings of National Academy of Science*. 81, 3088 – 3092.

Iwasokun, G. B., Egwuche, O. S., & Gabriel, J. A., 2015. Neural Network-Based Health Personnel Monitoring System. *African Journal of Computing & ICTs,* 81, 79 – 87.

Krishnamoorthy, C.S., & Rajeev, S., 1996. *Artificial Intelligence and Expert Systems for Engineers,* 198 – 218. CRC Press LLC.

Larose, D.T., 2005. *Discovering Knowledge in Data: An Introduction to Data Mining*, 60 – 214. New Jersey, U.S.A: John Wiley & Sons, Inc.

Larose, D.T., 2006. *Data Mining Methods and Models*, 204 – 239. New Jersey, U.S.A: John Wiley & Sons, Inc.

Lu, H., Setiono, R., & Liu, H., 1996. Effective Data Mining Using Neural Networks. In IEEE Transactions on Knowledge and Data Engineering. 8,6, 957 – 961.

Mitchell, M.T., 1997. Machine Learning, 81 – 127, 154 – 200. USA: McGraw-Hill Science/Engineering/ Math Publishing.

Osigbemeh, M.S., Ogunwolu, F.O., Omoare, A.A., & Inyiama, H.C., 2014. A Linguistic Fuzzy Expert System for Contagious Diseases Detection and Isolation. UNILAG Journal of Medicine, Science & Technology (UJMST), 2, 1 and 2, 1 – 10.

Pham, D.T., 1994. *Neural Networks in Engineering. In: Rzevski, G. et al. (Eds.), Applications of Artificial Intelligience in Engineering IX, AIENG/94, 3 – 36*. Proceedings of the 9th International Conf. on Computational Mechanics Publications, Southeamptom.

Rudin, C., & Wagstaff, K.L. 2013. Machine Learning for Science and Society. *Springer*. DOI 10.1007/s10994-013-5425-9

Rumelhart, D., & McClelland, J., 1986. *Parallel Distributed Processing*, 318 – 362. MIT Press, Cambridge.

Sebe, N., Cohen, I., Garg, A., & Huang, T.S., 2005. *Machine Learning in Computer Vision: Computational Imaging & Vision*, 187 – 244. Dordrecht-Netherlands: Springer

Suzuki, K., Shiraishi, J., Abe, H., MacMahon, H., & Doi, K., 2005. False- positive Reduction in Computer-aided Diagnostic Scheme for Detecting Nodules in Chest Radiographs by Means of Massive Training Artificial Neural Network. *Academic Radiology,* 12,2, 191 – 201. doi:10.1016/j.acra.2004.11.017.

Topping, B.H.V., Sziveri, J., Bahreinejad, A., Leite, J.P.B., & Cheng, B., 1998.Parallel Processing, Neural Networks and Genetic Algorithms. *Elsevier Science Ltd: Advances in Engineering Software*, 29,10, 763 – 786.

Van Rooji, A., Jain, L., & Johnson, R., 1996. *Neural Network Training Using Genetic Algorithms*. World Scientific, Singapore.

Wikipedia, 2015. *Artificial Neural Network.* Retrieved on 2nd of February, 2015 from https://en.wikipedia.org/wiki/Artificial_neural_network.

Wikipedia, 2016. *Confusion matrix*. Retrieved on 28th of April, 2016 from http://en.wikipedia.org/wiki/ Confusion_matrix

Zhou, Z., Chawla, N.V., Jin, Y., & Williams, G.J., 2015. *Big Data Opportunities and Challenges: Discussions from Data Analytics Perspectives*. IEEE Computational Intelligence Magazine. 9,4, 62 – 73