




Research Article

Comparison of the Artificial Neural Network's Approximations for the Levenberg-Marquardt Algorithm and the Gradient Descent Optimization on Datasets

Michael S. Osigbeme^{1*}, Chimereze Osuji¹, Moses O. Onyesolu² , Uche P. Onochie³

¹Department of Electrical and Electronics Engineering, Alex Ekwueme Federal University Ndufu-Alike, Nigeria

²Department of Computer Science, Nnamdi Azikiwe University, Awka, Nigeria

³Department of Mechanical Engineering, Delta State University of Science & Technology, Nigeria

E-mail: Michael.osigbeme@funai.edu.ng

Received: 16 October 2023; **Revised:** 4 February 2024; **Accepted:** 22 February 2024

Abstract: The approximations obtained by gradient descent optimization on a set of datasets were compared with the results obtained with the Levenberg-Marquardt Optimization Method (LMOM) on the same datasets. The datasets, which comprised three orthogonal databases obtained from MATLAB's Neural Network toolbox accompanying databases, were normalized and serially loaded to the artificial neural network Graphical User Interface (GUI) designed by the researchers. The GUI built with Visual Studio Programming Language (VSPL) implements a gradient descent optimization scheme of the back-propagation algorithm. The characteristics of each database for determination of the termination criteria were approximated from the developed feature extractive iteration algorithm. Revalidation sessions of the LMOM on the sampled datasets showed significant spuriousness in outputted results when compared with the gradient descent optimization results which although slow in attaining convergence produced results that can be closely replicated. Analysis of the F-statistics and the Receiver Operating Characteristics (ROC) for the sampled datasets results of both methods also showed that the gradient descent method demonstrated significant accuracy and parsimony in approximating the nonlinear solutions in the datasets when compared with the results from LMOM processing. Additionally, in this research, an algorithm for deducing and producing the ROC for analyzed Artificial Neural Network (ANN) sessions was also developed and implemented using VSPL.

Keywords: gradient descent optimization, Levenberg-Marquardt optimization, artificial neural network, back-propagation algorithm, iteration algorithm, receiver operating characteristic

1. Introduction

The avalanche increases in data acquisition systems and ease of creating databases have created the field of data mining. In addition, the dependence on machine learning and related algorithms that are based on nonlinear approximations has necessitated making sense of these huge databases. Machine learning principles such as artificial neural networks, fuzzy logic reasoning, genetic algorithms, and particle swarm optimization are presently increasing and are being used to interpret large datasets by connecting the logical dots embedded in such data sources.

Huge data mining has been used in healthcare for diagnosing a range of health challenges, including but not

limited to cancer detection, organ failure, cellular mutations, etc. Data mining has been used for fault detection and monitoring in automotive industries and critical systems such as space exploration and drilling. However, accompanying the widespread deployment of machine learning algorithms and their optimizations is the fact that their results and suggestions are still largely skeptical in usage and are only used in situations where there is no better alternative. Encouraging their deployments includes considerations of time, resources, the critical nature of the result, dataset size, etc. This dependence must be well thought out by the data mining engineer, and the perspective of the client or its application must be considered to ensure obtaining a logical solution and not a vague or ambiguous result of processing sessions.

The speed and accuracy of database mining using machine learning algorithms have improved in the past few decades with the evolution of various methods for speed enhancement while maintaining accuracy at all costs, at least for a justification of the use of such methods. The Levenberg-Marquardt Algorithm (LMA) has proven to be the fastest method for implementing the Back-Propagation Algorithm (BPA) of Artificial Neural Networks (ANNs) over the Gradient Descent Optimization (GDO) method. However, at what cost or sacrifice can the dependence on LMA only be allowed in its use in solving real-world problems based on machine learning optimizations? The nature of processed results from machine learning algorithms has suggested black box processing at least to the human perception of the analysis. Further compounding the black box concept of ANN-processed results, especially with the LMA, is the possibility of achieving a convergence solution in just one or a few iterations. Recall that though both GDO and LMA implement the Back-Propagation Algorithm (BPA), GDO uses the sum of the difference of squares method leading to slow convergences while LMA uses computation of the Jacobian matrixes or second-order methods to achieve BPA. Current methods of increasing the very slow speed of GDO, such as varying the learning rate in the gradient descent algorithm or introducing a momentum term in the GDO equation, though improved processing time have not achieved the speed of LMA's processing.

An attempt by this work to investigate and compare the accuracy of the ANN results of processing using the LMA analyzed with MATLAB Neural Network Toolbox and the result of the processing of GDO on the same dataset was carried out. The GDO scheme for implementing the back-propagation algorithm of the ANN and used for this comparison was developed by Osigbemeh et al. [1]. Insights into the development of the GDO method and its derivations can be obtained from Hagan et al. [2], Larose [3], Mitchell [4], etc. culminating in the present-day knowledge base of the method. The basis of this comparison was hinged on the lamentations by Nisbet et al. [5] on the apparent mistakes by data mining engineers in relying on one machine learning technique for database mining, especially for critical applications or tasks. Witten et al. [6] defined data mining as the automatic and semiautomatic process of discovering patterns in data such that the patterns discovered must be meaningful and lead to some advantage, usually an economic advantage. They also pointed out that with an avalanche of data that continues to increase, the proportion of comprehensible or understandable data by humans continues to decrease alarmingly. However, lying hidden in the data is information [5].

Attempts have been made by Kisi and Uncuoglu [7] to compare three back-propagation training algorithms on two case studies leading to an inference that the GDO was more resilient in their analysis when compared to LMA. Awolusi et al. [8] have used ANNs on steel fiber-reinforced concrete and the performance metrics of various topologies in training artificial neural networks have been investigated by Osigbemeh et al. [9].

Han et al. [10] and Aires et al. [11] noted that data mining methodologies should consider issues such as data uncertainty, noise, and incompleteness, while implementation should consider efficiency, scalability, performance, optimization, and the ability to execute in real time as key criteria that drive the development of many new data mining algorithms. Han et al. [10] also went further to itemize that data mining as a knowledge discovery process involves data cleaning, data integration, selection, transformation, pattern discovery, pattern evaluation, and knowledge presentation. Applications of these methodologies will no doubt allow for the possibility of data mining of huge databases without any significant challenges Zhou et al. [12].

The effect of database robustness on mining integrity was brought forward by Wilcox [13] by affirming that a database is robust if slight changes in a distribution have a relatively small effect on the values of the distribution. This when analyzing data helps to create an understanding of how robustness issues are addressed, and providing a reasonably good explanation requires some control theory. The issues of data connectivity in a database were investigated by Teorey et al. [14], who identified that when considering connectivity constraints, the connectivity

of a relationship describes a constraint on the connection of the associated entity in the relationship. They also gave guidelines for database modeling based on these entity relationships to include: clearly stating the database requirements before doing any entity relationship; modeling entities first, then relationships, and then attributes of relationships when appropriate; they also stated the identification of binary relationships when possible; keeping the conceptual model simple, according to a similar principle known as Occam's razor; and an often interaction with the end user or client to ensure that all parties share the same views and aims from the processed database. Additionally, supporting the use of moderate system architecture for building ANN is the proposition by Barron [15], Funahashi [16], and Suzuki et al. [17] for the use of three nodes in the hidden layer ANN architecture or topology as most times sufficient for implementing back-propagation and the use of local model ANN by Tzafestas [18].

2. Methodology

In this investigation, an experimental framework to compare LMA and the GDO method of data processing was deduced and experimented. The GDO method was used to iterate through each record in the different datasets named CancerDataset, ThyroidDataset, and WineDataset to produce the best or winning weights, which were then used to validate other records in each dataset. The independent datasets were obtained from the accompanying databases of Matlab's neural network toolbox. The cancer dataset contained 699 entries or records with 9 data input units to output to 2 units each, the thyroid dataset contained 7,200 entries in its database with 21 data input units to output to 3 units each and the wine dataset consists of 178 entries with 13 data input units to output to 3 units each. The determination of the winning weights was obtained by feeding random records in each of the three datasets into the ANN to attain a winning epoch or iteration from trial and error. Each unique epoch after determination from each dataset was used to calibrate or tune the other inputs to the ANN to maintain consistency. The weights for each session of determining the winning epoch, although generated randomly, were ensured to be optimal in getting the ANN into a unique winning epoch and were within the range of $0 < \text{weights} < 1$, i.e., values of fired weights were chosen to be between zero and one, such that each weight was different from the others but within the same acceptable boundaries. For each session of the investigated dataset, the sensitivity of the ANN performance was investigated by determining the F-statistic test and the Receiver Operating Characteristic (ROC) curve. A ROC is a graphical curve originally used for analysis and interpretation of the characterization and trade-off between hit rate and false-alarm rate over a noisy channel in signal detection and analysis. The F-statistic was used to produce the confusion matrix table shown in the figures accompanying the GDO processing results.

2.1 Data preparation and visualization

Since one of the objectives of this work is to compare and validate the LMA's operation or processing of the various datasets with GDO processing, it was necessary for each dataset to be inspected for possible normalization and consistency to improve the ANN processing of the data. This is also hinged on the emphasis by Swets [19] that in diagnosing systems, the determination of optimal solution for solving resulting problems that are classifiable as signals and noise may have sound measures (or concepts), for which "the values obtained from tests of (the) diagnostic systems often require qualification because the test data on which they are based are (mostly) of unsure quality". The Cancer database was used since it conformed to the ANN and required a data range of cell values of 0.0 to 0.99 (less than 1). However, the other datasets used for this analysis, i.e., Wine and Thyroid datasets were normalized based on the following mathematical operations.

2.2 Data normalization

To obtain a logical input of data into the ANN for each iteration, the data in the dataset were normalized using the following scheme outlined below. Since each dataset of the three databases used in the analysis was generated or obtained separately with different and unique characteristics, each column value had to be normalized to be accommodated by the gradient descent optimization architecture used in designing the ANN. For the Thyroid database, rows that had values in the range of 0.000X, where X is an integer multiplied by 1,000 and the sigmoid function was

further used to normalize the data to conform to a value of less than 1, i.e.,

$$\text{New cell value} = 1,000 \times 1/(1 + (e - \text{Current cell value})) \quad (1)$$

Additionally, rows that were in the range of 0.00X were normalized with the syntax below

$$\text{New cell value} = 100 \times 1/(1 + (e - \text{Current cell value})) \quad (2)$$

The following functions were used to normalize the Wine database-cell rows that contained cell values greater than 1:

$$\text{New cell value} = 1/(1 + (e - \text{Current cell value}/10)) \quad (3)$$

$$\text{New cell value} = 1/(1 + (e - \text{Current cell value}/1,000)) \quad (4)$$

Values of row cells with less than 1 in the Wine database were normalized with

$$\text{New cell value} = 1/(1 + (e - \text{Current cell value} \times 10)) \quad (5)$$

This normalization provided a treatment of imprecision and uncertainty in the three orthogonal or different datasets for better interpretability and processing by the ANN. Since there are few or no descriptive or explanatory features and properties of the considered datasets, a dataset parsing algorithm was developed and is shown in Table 1. The developed feature extractive iteration algorithm helps to determine the optimal training and validation epoch or iteration needed for the ANN to attain convergence. This is done by considering a “camera glance” of the data in the dataset. By camera glance, the algorithm seeks to use a matrix optimization approach to determine winning or considerable data columns for the unknown database features and run the specified columns data or samples through the ANN to obtain convergence. These convergence data were then used as training features for the rest of the sub-dataset, i.e. The iteration algorithm was used for ANN processing after prior separation of the dataset into 70% training data, 15% validation, and 15% testing data.

However, the iteration algorithm is essentially a heuristic process that attempts to recognize pattern consistency using random sampling of specific samples in the dataset to create a convergence epoch based on trial and error. In multiclass data, such as the WineDataset and the ThyroidDataset, which had three classes of outputs and targets, the iteration algorithm was modified to recognize the output targets and their corresponding inputs as unique data in the sampled datasets.

Data normalization is also a step towards demystifying the black box concept of artificial neural processing for real-world databases to ensure consistent and reverifiable ANN processing.

3. Results and discussion

To obtain an indication of the efficiency and sensitivity of ANN performance on the LMA and the GDO, a comparison of the two optimization methods was carried out with the same datasets, the same topological architecture of using three nodes in the hidden layer, the same percentage of training samples, validation and testing samples and with the same serial loading of all data in the datasets. However, the following specific results were obtained from their comparison. The GDO method that was used for this comparison showed the following results: the experimentally determined epochs for each dataset after normalization yielded 160 epochs for the CancerDataset; a 170 epoch convergence epoch on the ThyroidDataset was attained and 150 epochs on the WineDataset. A learning rate (η) of 0.3; $0 < \eta < 1$ as an arbitrary constant was chosen to help obtain the generated neural network weights faster towards its global minimum for the Sum of Square Errors (SSE) while still preserving the integrity of the solution of the GDO method. The generated weight choice was chosen such that they were randomly generated in each session and within a suitable

range of 0.1 to 0.9. The following results were obtained by processing the three different datasets.

Table 1. The feature extractive iteration algorithm

Algorithm 1
<p>Load all samples in the dataset Read Target Class Read dataset Inputs 1. Initialization: Read the size of the dataset (Ψ_n). The size corresponds to the last column or upper bound n^{th} cell. Obtain Ψ_1 - the whole number of the division of size by two, i.e., $\Psi_{n/2}$. Obtain Ψ_2 - the whole number of divisions of Ψ_1 by two, i.e., $\Psi_{1/2}$. Obtain Ψ_a - the whole number of divisions of Ψ_n by four, i.e., $\Psi_{n/4}$. Obtain Ψ_3 - the whole number of divisions of Ψ_2 by two, i.e., $\Psi_{2/2}$. Obtain Ψ_b - the whole number of divisions of Ψ_n by eight, i.e., $\Psi_{n/8}$. Obtain Ψ_i - the whole number of Ψ_2 additions with $\Psi_{n/4}$. Obtain Ψ_{ii} - the whole number of additions with $\Psi_{n/8}$.</p> <p>2. Loop: for large database or highly inconsistent database With Ψ_{0+1} as new Ψ_0 and Ψ_{n+1} as Ψ_n Determine $\Psi_{x(\text{new})}$ Next Ψ</p> <p>3. Update: loading data to ANN to determine convergence epoch Load Ψ_0 - the first column data samples Iterate and obtain convergence epoch Then Load data samples corresponding to the position of Ψ_x on the dataset where x is the x^{th} column number; i.e., $\Psi_n, \Psi_1, \Psi_2, \Psi_3, \Psi_i, \Psi_{ii}, \Psi_a,$ and Ψ_b Obtain convergence epoch for each Ψ_x Repeat step 2 to obtain fine-tuned features if the database is large</p> <p>4. Visualization: Obtain the average of the converged epochs of each extracted feature from the outputted classes to be used for ANN training</p> <p>5. Training: If the training requirement is satisfied Then, Stop Else, Go to 2</p>

3.1 Experiments with the GDO method on Cancer Data

The CancerDataset with the determined convergence epoch as 160th generated the confusion matrix [20-22] of all iteration sessions shown in Figure 1. The CancerDataset, which contained 699 samples, was collated from data obtained from patients based on the classification of breast cancer as either benign or malignant depending on the characteristics of the sampled patient biopsies. The table shows 395 (56.5%) data samples present in the predicted condition of the first class and 221 (31.6%) of the true negatives of that class. False positives constitute approximately 22 (3.1%) data samples of the second class, with a true positive outcome of 61 constituting approximately 8.7% of the data. Since the analysis was based on the database obtained without prior information on which of the samples are benign and which are malignant, the results here cannot point out which is which except for additional information from the client end

or samples collection source; however, one class must be benign and the other class malignant. The ROC graph in Figure 2, which was obtained from [20-21] and implemented using the Visual Studio Programming Language (VSPL), graphically shows the distribution of the data samples as they perform with the gradient descent method. The algorithm that was used in implementing the ROC curve for this analysis is shown in Table 2. A network topology of nine input units representing the biopsy sample characteristics, three hidden layers for ANN operation, and two output units for ANN to converge to and visualize the data. Seventy percent of the data set was used as training, 15% as validation, and another 15% as testing data for the ANN. A topological illustration of the ANN architecture for the GDO is shown in Figure 3 and the VSPL used was the Visual Basic 6.0 programming language.

Table 2. The algorithm for creating the gradient descent ROC

Algorithm 2	
	Load all iteration sessions
	Read Target class
	Read Output class
	Initialize:
	Read the first epoch for true positive for class 1
	Determine convergence to x as x_1 (corresponding to true positives)
	Continue until convergence to y
	Read convergence to y at y_1 (corresponding to true negatives)
	Continue until convergence to x
	Plot ($x_1, x_2, \dots, x_n, y_1, y_2, \dots, y_n$)
	Repeat until the last epoch
	Update:
	Plot (X, Y)
	Complete plot
	Repeat the process for true positives for class 2
	Complete plot
	Repeat the process for true positives for the nth class
	Complete plot
	End

		1		
		395	61	86.6%
Output	1	56.5%	8.7%	13.4%
	2	22	221	90.9%
		2		
		3.1%	31.6%	9.1%
Output	1	94.7%	78.4%	88.1%
	2	5.3%	21.6%	11.9%
		1	2	
		Target		

Figure 1. Confusion matrix for CancerDataset

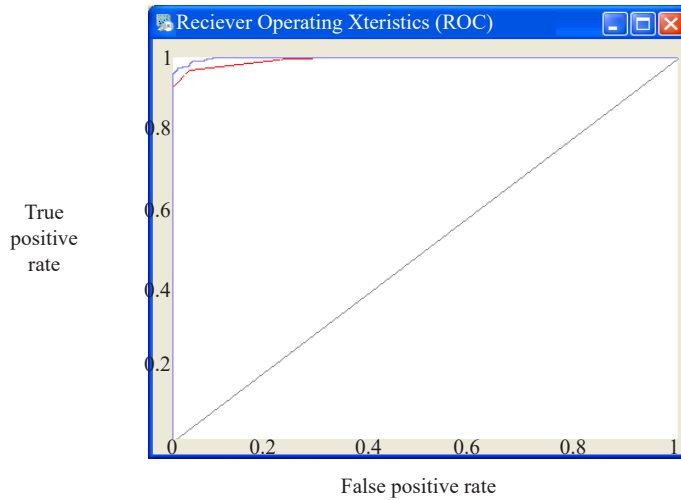


Figure 2. ROC for CancerDataset

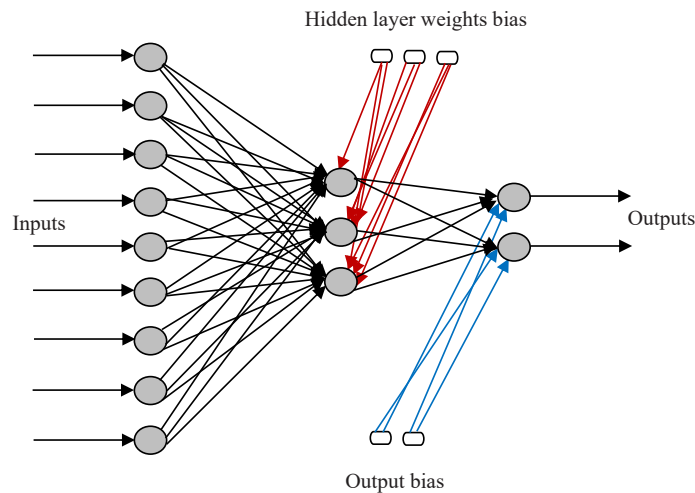


Figure 3. Topology of ANN for GDO of CancerDataset

3.2 Experiments with the LMA optimization method on Cancer Data

The results of running MATLAB's neural network toolbox that implements the LMA on the same datasets were as follows: the 699 samples CancerDataset were divided into 70% training data, 15% validation, and 15% testing data in approximately 22 iterations, producing the confusion matrix and ROC curve shown in Figure 4 and 5. A network topology of nine input units, three hidden layers, and two output units was also used in the experiments. The network processing, however, suffered from spurious or different solutions based on unsuitable generated weights used in the iterations by the MATLAB neural network toolbox, which converses with the operation of the GDO and will produce a more guaranteed or stable solution that is devoid of large changes in revalidation. With repetitive runs, the best or optimal solution for the LMA convergence sessions and processing obtained the results shown in Figure 4 and the corresponding ROC in Figure 5.

All Confusion Matrix

Output Class	1	446 63.8%	11 1.6%	97.6% 2.4%
	2	12 1.7%	230 32.9%	95.0% 5.0%
		97.4% 2.6%	95.4% 4.6%	96.7% 3.3%
		1	2	
		Target Class		

Figure 4. Confusion matrix for CancerDataset

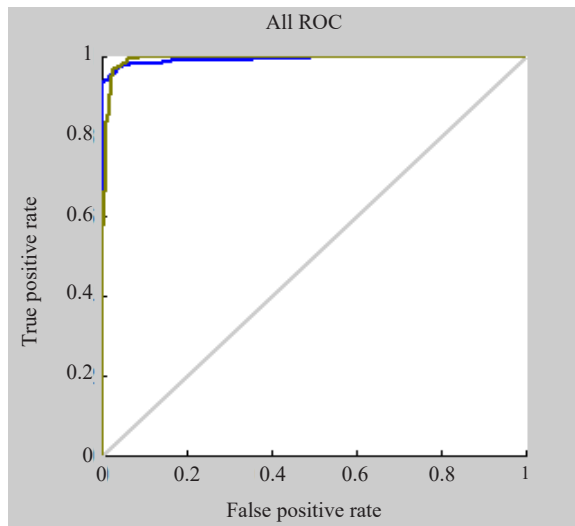


Figure 5. ROC for CancerDataset

3.3 Experiments with the GDO method on Wine Data

The WineDataset with the experimentally determined convergence epoch at the 150th generated the confusion matrix of all iteration sessions shown in Figure 6. The dataset, which contained 178 results of sample tests of three Italian wineries based on each winery's wine constituents obtained through prior chemical analysis, was used to obtain the results from GDO processing. A network topology of thirteen (13) input units, three hidden layers, and three output units or target classes was used in the experiments. The results show that 33.1%, representing 59 sample tests of the true positives for class 1, was obtained, with 37.1%, representing 66 test samples as true positives for class 2 and a corresponding 27% or 48 test samples as true positives for class 3. False negatives for class 1 and true negatives for class 3 were also observed, as shown in Figure 6. The ROC curve shown in Figure 7 shows almost smooth graph symmetry due to the few false negatives in the processed data. However, it must be recalled that participating datasets have been normalized according to the normalization equations of section 2.2 to be able to pass the datasets through the designed ANN.

All data samples confusion matrix

Output	1	59 33.1%	3 1.7%	0 0.0%	95.2% 4.8%
	2	0 0.0%	66 37.1%	0 0.0%	100.0% 0.0%
	3	0 0.0%	2 1.1%	48 27.0%	96.0% 4.0%
		100.0% 0.0%	93.0% 7.0%	100.0% 0.0%	97.2% 2.8%
		1	2	3	
		Target			

Figure 6. Confusion matrix for WineDataset

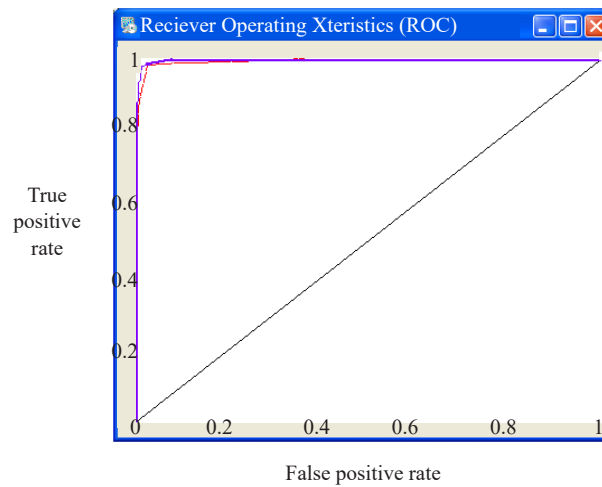


Figure 7. ROC for WineDataset

3.4 Experiments with the LMA optimization method on Wine Data

The LMA method processing produced the following results on the WineDataset, which contained 178 data samples and was also divided into 70% training data, 15% validation, and 15% testing data. The dataset in approximately 19 iterations using the same network topology of the GDO (i.e., thirteen input units, three hidden layers, and three output units) produced the confusion matrix and ROC curve in Figure 8 and 9, respectively. It must be emphasized that the LMA method produces slightly different results for different iterative sessions when revalidated, and the results presented here are based on the closest optimal results obtainable from experimental runs.

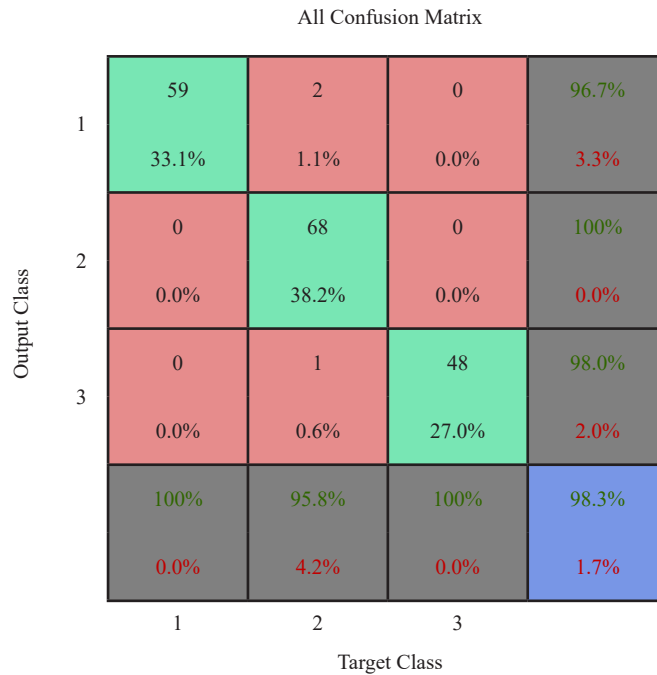


Figure 8. Confusion matrix for WineDataset

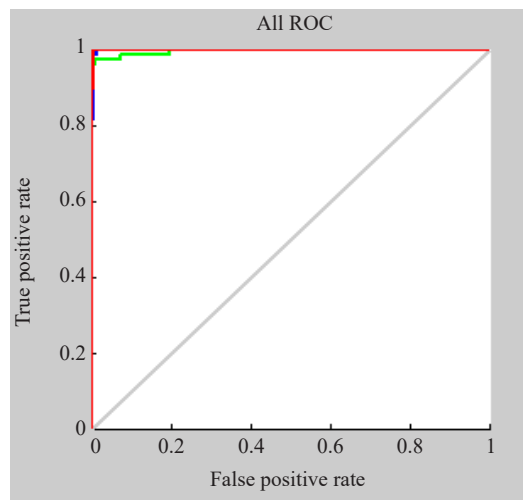


Figure 9. ROC for WineDataset

3.5 Experiments with the GDO method on Thyroid Data

The ThyroidDataset contained three orthogonal classes for artificial neural network determination and convergence. This led to an experimentally determined convergence epoch at the 170th epoch with a preset learning rate (η) of 0.3; $0 < \eta < 1$ was chosen to help obtain the generated neural network weights faster towards its global minimum for the Sum of Square Errors (SSE) while still preserving the generalizability of the GDO solution on new data. The dataset under analysis represented three target classes of a clinic's categorization of patients' presented conditions as either normal, subnormal, or hyper-functional states during diagnostic sessions. A network topology of twenty-one input units, three hidden layers, and three output units or target classes was used in the experiments. In the large dataset, 7,200 sampled

patients were also divided into 70% training, 15% validation, and 15% testing data to obtain the results shown in Figure 10 and 11.

Output	1	152 2.1%	20 0.3%	19 0.3%	79.6% 20.4%
	2	0 0.0%	0 0.0%	0 0.0%	#DIV/0! #DIV/0!
3	14 0.2%	356 4.9%	6,639 92.2%	94.7% 5.3%	
	91.6% 8.4%	0.0% 100.0%	99.7% 0.3%	94.3% 5.7%	
		1	2	3	
		Target			

Figure 10. Confusion matrix for ThyroidDataset

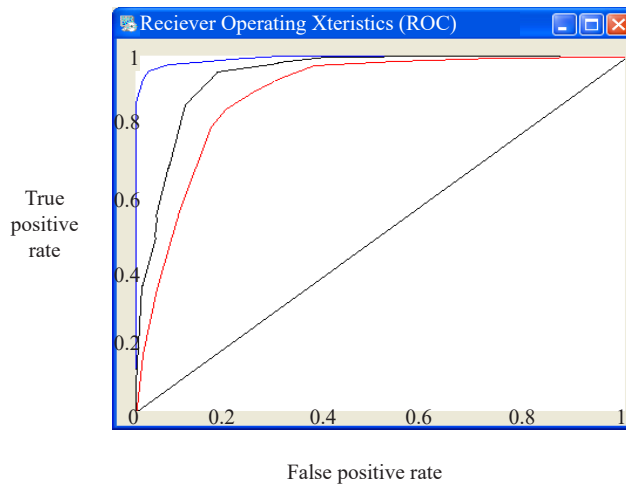


Figure 11. ROC for ThyroidDataset

The confusion matrix of Figure 10 shows that GDO processing returned a true positive of 152 samples representing 2.1% of classes 1 and 20, with 19 sampled for classes 2 and 3 contributing a total of 0.6%. The second class, which corresponds with patients categorized as subnormal, showed that the dataset did not contain entries to this categorization of the patient’s condition, thus returning a null on the confusion matrix. These null entries, which could be traced to man-made errors in obtaining or inputting data or data corruption, generated the error #DIV/0! in the table by the Microsoft Excel package that was used to obtain the matrix. The majority of the sampled data were localized in class 3; the output-target class of the matrix corresponded to approximately 92.2% of that class. Fourteen (0.2%) samples tested

positive for class 1, while 356 (4.9%) tested positive for class 2. The generated ROC for the ThyroidDataset is shown in Figure 11, and it clearly shows the distinct characterization of the various data sources for the three different entries for the patient's presented symptoms.

All Confusion Matrix

Output Class	1	144 2.0%	26 0.4%	21 0.3%	75.4% 24.6%
	2	0 0.0%	0 0.0%	0 0.0%	NaN% NaN%
	3	22 0.3%	342 4.8%	6,645 92.3%	94.8% 5.2%
		86.7% 13.3%	0.0% 100%	99.7% 0.3%	94.3% 5.7%
		Target Class			
		1	2	3	

Figure 12. Confusion matrix for ThyroidDataset

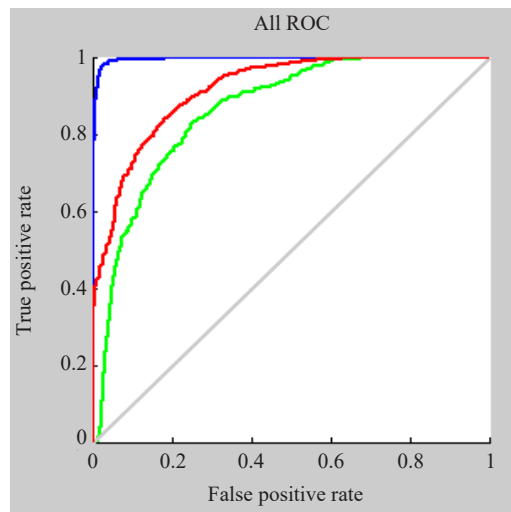


Figure 13. ROC for ThyroidDataset

3.6 Experiments with the LMA optimization method on Thyroid Data

Additionally, the LMA method produced the following on the ThyroidDataset which also contained 7,200 data samples and was also divided into 70% training data, 15% validation, and 15% testing data and produced the confusion

matrix table and ROC curve in Figure 12 and 13 after approximately 46 iterations. This network topology of twenty-one input units, three hidden layers, and three output units to accommodate the three target classes was also used in the experiments. The dataset represented the three classes of targets corresponding to the clinic’s classification of patients as normal, hyper-functional, or subnormal functioning. A network topology of twenty-one input units, three hidden layers, and three output units or target classes was also used in the experiments. Several validation attempts showed that the results of processing using the LMA only produced spurious results in several runs.

3.7 Data analysis

The analysis of these datasets has shown the various merits and demerits of the two methods under comparison with LMA, demonstrating unmatchable and superior execution times but with appreciable distortions in revalidation results. GDO, on the other hand, has demonstrated a more consistent solution that is closer to the natural problem-solving or approximation ability of humans. In GDO processing, appreciable control was made available to the user, as much of the operation depends on user inputs in the determination of the optimum range of weights, appropriate learning rate, optimal convergence epoch given the best or winning weights, and termination criteria. This is mainly because sequential execution of machine learning is still the best approach to obtaining approximations for nonlinear problems with improvement in time of execution obtained from a careful adjustment of learning rate and momentum term in the GDO equation. This feature for increasing iteration time in GDO has helped to reduce the “black box” concept of this method of machine learning when compared to LMA, which considers datasets to be more mathematical matrices than more real data samples. Although MATLAB’s execution of the LMA method ensured speed and quick data convergence in just a few iterations, more user flexibility and control were provided by the traditional GDO for this set of datasets, leading to more interpretable results. In this comparison of both methods with the same datasets and topologies, all methods were carried out in accordance with relevant guidelines and regulations.

3.7.1 Summary of the techniques on each dataset

With the inability to revalidate the execution by the LMA method of processing and coupled with inconsistencies in the number of iterations to achieve convergence, thereby creating spuriousness in the analysis of ANN and black-box concept, Table 3 shows these comparisons with the GDO results of analysis on same datasets and architectural topologies. Also, Table 4 shows the user’s interactivenss of GDO analysis with the ability for adjustable learning rate for speed control and momentum term control over the LMA operation’s speed.

Table 3. Performance of both methods on the various classes of datasets

Dataset	ANN method					
	LMA			GDO		
Cancer	Fast execution	Quick convergence	Opaque	Serial execution	Slow convergence	Easily interpretable
Thyroid	Fast iterations	Jacobian matrix based	Inconsistent revalidation	Slow iterations	The sum of squares based	Easy revalidation
Wine	Fewer computation resources	Unreliable solution	No user control terms	More computer resources	Amenable to a real solution	Learning rate, momentum term control

3.7.2 Numerical summary of the techniques on each dataset

Table 4. Performance metrics of both methods on the same datasets

Dataset	ANN method					
	No of Epoch	LMA Learning Rate	Av. Training Time	No of Epoch	GDO Learning Rate	Av. Training Time
Cancer (699)	22	NA	16 Secs	160	0.3	2½ Mins
Thyroid (7,200)	46	NA	50 Secs	170	0.3	3 Mins
Wine (178)	19	NA	11 Secs	150	0.3	2 Mins

NA: Not available

4. Conclusion

The role of data mining for feature extraction of large databases has been investigated in this work. The outcome of approximations obtained from the LMA should be used with great caution and intuition, as repeated iteration sessions with this method only have produced different results of sampling datasets used for its revalidation. This is of utmost importance when considering the real-world application of the method in machine learning. The response of the same dataset by the gradient descent method produced a guaranteed convergence at an appreciable number of epochs, and an increase in the time for the peculiar slow convergence of GDO was improved by setting the learning rate to a value of 0.3 while still ensuring minimal error during iterations. The datasets used for the comparison were normalized and tested under the same conditions, such as the same neural network topology, number of input and output target pairs, same order of loading samples, and same percentage of training, validation, and testing sets. The developed iteration algorithm attempted to recognize pattern consistency using random sampling of specific samples in the dataset to create a convergence epoch heuristically. The ROC graph for each session was obtained from an implementation of the developed algorithm using visual studio programming and has shown a significant representation of the iterative results of ANN processing for both methods. For time-dependent solutions, the LMA could be used symbiotically to first obtain a result for further application or processing while awaiting the results of iterations using the inherently slow but stable GDO.

Conflict of interest

The authors declare that they have no conflicts of interest.

References

- [1] Osigbeme MS, Ohaneme CO, Inyama HC. An algorithm for characterizing prefuzzified linguistic nuance using neural network. *Springer International Journal of Speech Technology*. 2017; 20(2): 355-362. Available from: doi: 10.1007/s10772-017-9413-5.
- [2] Hagan MT, Demuth HB, Beale MH, DeJesus O. *Neural Network Design*. 2nd ed. Boston: PWS Publishing; 2013.
- [3] Larose DT. *Data Mining Methods and Models*. New Jersey: John Wiley & Sons, Inc.; 2006. p. 204-239.
- [4] Mitchell MT. *Machine Learning*. Boston: McGraw-Hill Science/Engineering/Math Publishing; 1997. p. 81-127, 154-200.
- [5] Nisbet R, Elder J, Miner G. *Handbook of Statistical Analysis and Data Mining Applications*. Burlington, Massachusetts: Academic press-Elsevier Inc. Burlington; 2009.
- [6] Witten IH, Frank E, Hall MA. *Data Mining: Practical Machine Learning Tools and Techniques*. 3rd ed. Burlington,

Massachusetts: Morgan Kaufmann Publishers-Elsevier Inc.; 2011.

- [7] Kisi O, Uncuoglu E. Comparison of three back-propagation training algorithms for two case studies. *Indian Journal of Engineering and Materials Science*. 2005; 12: 434-442.
- [8] Awolusi TF, Oke OL, Akinkulore OO, Sojobi AO, Aluko OG. Performance comparison of neural network training algorithms in the modeling properties of steel fiber reinforced concrete. *Heliyon*. 2019; 5(1): e01115. Available from: doi: 10.1016/j.heliyon.2018.e01115.
- [9] Osigbeme MS, Okezie CC, Inyama HC. Performance metrics of various topologies of a feed forward error-back propagation neural network. *Journal of Engineering and Applied Science (JEAS)*. 2017; 12(10): 94-102.
- [10] Han JW, Kamber M, Pei J. *Data Mining: Concepts and Techniques*. 3rd ed. Burlington, Massachusetts: Morgan Kaufmann Publishers-Elsevier Inc. Waltham; 2012.
- [11] Aires F, Prigent C, Rossow W. Neural network Uncertainty assessment using bayesian statistics: A remote sensing application. *Neural Computation*. 2004; 16(11): 2415-2458.
- [12] Zhou Z, Chawla NV, Jin Y, Williams GJ. Big data opportunities and challenges: discussions from data analytics perspectives. *IEEE Computational Intelligence Magazine*. 2015; 9(4): 62-73. Available from: doi: 10.1109/MCI.2014.2350953.
- [13] Wilcox RR. *Introduction to Robust Estimation and Hypothesis Testing*. 2nd ed. Burlington, Massachusetts: Elsevier Inc.; 2005.
- [14] Teorey S, Lightstone S, Nadeau T, Jagadish HV. *Database Modeling and Design: Logical Design*. 5th ed. Burlington, Massachusetts: Morgan Kaufmann Publishers-Elsevier Inc.; 2011.
- [15] Barron AR. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transaction on Information Theory*. 1993; 39(1): 930-945. Available from: doi: 10.1109/18.256500.
- [16] Funahashi K. On the approximate realization of continuous mappings by neural networks. *Neural Networks*. 1989; 2(3): 183-192. Available from: doi: 10.1016/0893-6080(89)90003-8.
- [17] Suzuki K, Shiraishi J, Abe H, MacMahon H, Doi K. False-positive reduction in computer-aided diagnostic scheme for detecting nodules in chest radiographs by means of massive training artificial neural network¹. *Academic Radiology*. 2005; 12(2): 191-201. Available from: doi: 10.1016/j.acra.2004.11.017.
- [18] Tzafestas SG, Skoundrianos EN, Rigatos GG. Nonlinear neural control of discrete time systems using local model neural network. *International Journal of Knowledge-Based Intelligent Engineering Systems*. 2002; 4(2): 130-140.
- [19] Swets J. Measuring the accuracy of diagnostic systems. *Science*. 1988; 240(4857): 1285-1293.
- [20] Fawcett T. *ROC Graphs: Notes and Practical Considerations for Data Mining Researchers*. Palo Alto, California: Hewlett-Packard Company; 2003.
- [21] Fawcett T. An introduction to ROC analysis. *Elsevier Pattern Recognition Letters*. 2005; 27(8): 861-874. Available from: doi: 10.1016/j.patrec.2005.10.010.
- [22] Wikipedia. *Confusion matrix*. 2016. Available from: http://en.wikipedia.org/wiki/Confusion_matrix [Accessed 28th April 2016].