Input Neuron Prediction Based on Error Minimization of Percentage Proportion of Hidden Neurons: A Comparative Study of Levenberg-Marquardt and Scaled Conjugate Gradient Algorithms

Nebo Ephraim Ugochukwu [1], Onah Okechukwu Thomas [2], Aka Christian Chikezie [3*]
[1] Department of Mechatronics Engineering, Faculty of Engineering, Enugu State University of Science & Technology, Enugu, Nigeria
[2] Department of Mechanical Engineering, Faculty of Engineering, Federal College of Agriculture Isiagu, Ebonyi State, Nigeria

* Corresponding Author: **Aka Christian Chikezie**

Article Info

P-ISSN: 3051-3383
E-ISSN: 3051-3391
Volume: 06
Issue: 02
July - December 2025
Received: 18-10-2025
Accepted: 21-11-2025
Published: 15-12-2025
Page No: 179-190

Abstract
This study investigates input neuron prediction based on the error minimization outcomes of percentage proportion of hidden neurons using two prominent training algorithms: Levenberg-Marquardt (LM) and Scaled Conjugate Gradient (SCG). The research addresses the critical challenge of uncertainty minimization in neural network predictions by systematically evaluating various training state proportions (80%, 70%, 60%, 50%, 40%, and 30%) and hidden neuron configurations (2, 4, 6, 8, 10, 20, 30, 40, and 50). The validation performance analysis reveals that the Levenberg-Marquardt algorithm achieves optimal results at 80% training proportion with an error value of 8.194×10−7 at epoch 363, significantly outperforming the Scaled Conjugate Gradient approach. Error histogram analysis indicates that 60% training proportion yields the least error distribution for both methods, establishing this as the optimal configuration for error histogram minimization. Furthermore, hidden neuron analysis demonstrates that 8 neurons produce the best validation performance with minimal error of 0.00012831 at epoch 94, while configurations approaching or exceeding 40-50 neurons exhibit underfitting characteristics with dispersed error histograms. The comparative study conclusively establishes the Levenberg-Marquardt algorithm's superiority in input neuron prediction for neural network training, providing valuable guidelines for practitioners in medical diagnostics, industrial automation, and traffic prediction applications.

DOI: https://doi.org/10.54660/IJAIET.2025.6.2.179-190

Keywords: Neural Network, Matlab Simulation, Mathematical Model, Optimization, Algorithm

1. Introduction

Artificial Neural Networks (ANNs) have emerged as powerful computational methodologies that enable machine learning, knowledge demonstration, and the application of learned knowledge to maximize the output responses of complex systems [1, 2]. The architecture of artificial neural networks mimics that of the human brain, with web-like connections between neuronal nodes. Each neuron serves as a building block that transmits and receives information through axons and dendrites, with synaptic connections that can undergo variations in strength when exposed to external stimuli [3, 4].

Pattern recognition, including line recognition, speech recognition, and image processing, along with classification tasks such as text or image classification, represent the many modern applications of neural networks [5]. These networks are utilized as actuators in pattern recognition within engineering robotics and mechatronics systems [6], with additional applications in energy production systems, particularly wind energy, for improving efficiency through wind speed and direction prediction [7, 8].

The training of neural networks involves various algorithmic techniques to minimize errors caused by underfitting or overfitting. Among these algorithms, the Levenberg-Marquardt (LM) and Scaled Conjugate Gradient (SCG) algorithms have demonstrated significant potential for uncertainty reduction in neural network predictions [9, 10]. However, certain neural network algorithms do not converge quickly, do not achieve higher performance, and do not attain the best accuracy during testing, necessitating a comprehensive comparative study of these approaches.

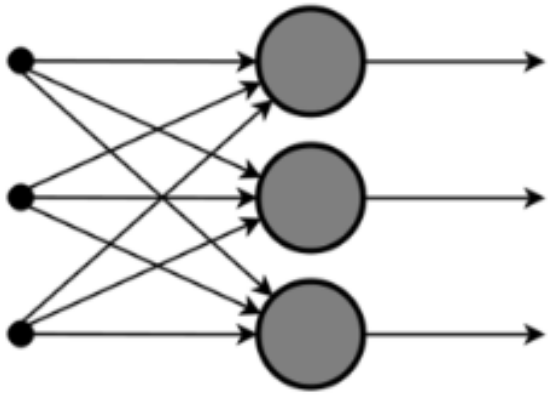


Fig 1: Basic architecture of an Artificial Neural Network (ANN) showing the interconnections between input layer, hidden layer, and output layer [10].

The primary objective of this research focuses on the prediction of input neurons based on the outcome of error minimization of percentage proportion of hidden neurons using both the Levenberg-Marquardt and Scaled Conjugate Gradient algorithm approaches. This objective addresses the critical need to identify optimal training configurations that minimize uncertainty and improve prediction accuracy in neural network applications.

1.1. Problem Statement

Neural networks employ various algorithmic techniques during training and retraining processes to minimize errors caused by underfitting or overfitting. The challenge lies in determining the optimal percentage proportion of training states and hidden neuron configurations that achieve the best validation performance while minimizing prediction errors. Current literature lacks comprehensive comparative analysis of how different training proportions (80%, 70%, 60%, 50%, 40%, and 30%) and hidden neuron configurations affect input neuron prediction accuracy using both LM and SCG algorithms.

1.2. Research Objectives

The specific objective addressed in this study is the prediction of input neurons based on the outcome of error minimization of percentage proportion of hidden neurons/default of the two methods. This involves: (a) analyzing validation performance across different training proportions, (b) examining error histogram distributions to identify optimal configurations, (c) evaluating regression analysis for prediction accuracy, and (d) comparing the efficacy of LM and SCG algorithms for input neuron prediction.

2. Literature Review

2.1. Neural Network Training Algorithms

The conjugate gradient approach, as described by Hestenes and Stiefel [11], allows separate line searches to each search direction by constructing a set of conjugate search directions under the assumption of a quadratic error surface. By conducting a line search in every iteration to determine the optimal step size error (α) along the conjugate direction, this approach increases the training procedure's complexity while providing systematic convergence guarantees.

The Scaled Conjugate Gradient method, developed by Møller [12], finds the best step size at each iteration using a Levenberg-Marquardt approach rather than a line search. This modification significantly reduces computational overhead while maintaining convergence properties. The Levenberg-Marquardt algorithm, originally developed for numerical least-squares non-linear function minimization [13, 14], has found extensive application in reducing neural networks' error functions, as documented by Hagan and Menhaj [15].

2.2. Levenberg-Marquardt Algorithm Theory

The Levenberg-Marquardt technique derives from considering the error E following a differential shift in the ANN weights from θ₀ to θ according to the second-order Taylor series [16] as in Eq. (1).

$$E(\theta) = E(\theta_0) + g^T(\theta - \theta_0) + \frac{1}{2}(\theta - \theta_0)^T \mathrm{H}(\theta - \theta_0) + H.O.T. \quad (1)$$

where g = ∇E(θ) is the gradient vector and H represents the Hessian matrix containing the second partial derivatives of the loss function. The Levenberg-Marquardt learning algorithm modifies the Gauss-Newton approach as in Eq. (2).

$$\theta_{i+1} = \theta_i - \eta_i \mathrm{H}_i^{-1} g_i = \theta_i - \frac{1}{2}\eta_i \left(J_i^T J_i + \lambda_i I\right)^{-1} g_i \quad (2)$$

where λ is a scalar damping parameter and I is the identity matrix. This algorithm reduces to Gauss-Newton as λ approaches zero and to steepest descent in the limit as λ approaches infinity. The learning rate η and factor λ are designed to change adaptively during training, typically starting with large values and gradually decreasing as the solution improves [17].

2.3. Overfitting and Underfitting in Neural Networks

An overfitting problem exists in supervised machine learning when a model fails to adequately generalize from observed data to unseen data [18]. Overfitting causes the model to fit poorly to the testing set even when it performs well on the training set, as the over-fitted model has difficulty responding to differences between training and testing data points. Over-fitted models tend to memorize the training set's inevitable noise rather than discovering the hidden patterns in the data [19].

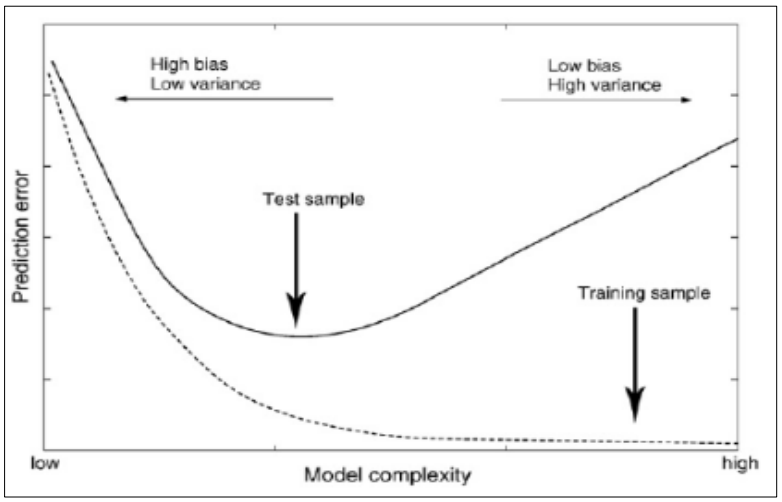


Fig 2: Validation error versus testing error [19].

Figure 2 illustrating the bias-variance tradeoff in neural network training. The optimal stopping point occurs where validation error is minimized before overfitting begins.
The epoch is depicted on the horizontal axis and the error on the vertical axis in Fig. 2. If the model continues to learn beyond the optimal point, the validation error will increase while the training error continues to decrease. This phenomenon necessitates a technique based on the Scaled Conjugate Gradient algorithm and the Levenberg-Marquardt predictor for neural networks to minimize uncertainty [20, 21].

2.4. Hidden Neuron Configuration Effects

The selection of appropriate hidden neuron configurations significantly impacts neural network performance. Too few hidden neurons may result in underfitting, where the network cannot capture the underlying patterns in the data. Conversely, too many hidden neurons may lead to overfitting, where the network memorizes training data including noise [22]. The optimal number of hidden neurons depends on the complexity of the problem, the size of the training dataset, and the desired generalization capability [23].

3. Materials and Methods

3.1. Experimental Setup

The experiments were conducted using MATLAB R2019a with the Neural Network Toolbox on a laptop computer. The neural network was configured with a default value of 10 hidden neurons and 70% training proportion. Systematic variations were applied to both the training proportions (80%, 60%, 50%, 40%, and 30%) and hidden neuron configurations (2 and 50, 4 and 40, 6 and 30, and 8 and 20) to comprehensively evaluate the prediction performance.

3.2. Data Division Strategy

The dataset consisting of 94 samples was divided according to the following default proportions shown in table 1.

Table 1: Default data division configuration for neural network training.

Data Set	Percentage	Number of Samples	Purpose
Training	70%	66	Network weight adjustment based on error
Validation	15%	14	Measure generalization, halt when stops improving
Testing	15%	14	Independent measure of network performance

3.3. Training Algorithm Configuration

Two training algorithms were employed for comparative analysis:

Levenberg-Marquardt Algorithm (trainlm): This algorithm typically requires more memory but less time. Training automatically stops when generalization stops improving, as indicated by an increase in the mean squared error of the validation samples. The algorithm uses data division by random selection (dividerand) and employs mean squared error (mse) as the performance metric.

Scaled Conjugate Gradient Algorithm (trainscg): This algorithm uses the Levenberg-Marquardt approach to find the optimal step size at each iteration rather than performing a line search. It offers lower memory requirements compared to LM, making it suitable for larger network architectures.

3.4. Network Architecture

A fitting neural network (fitnet) architecture was implemented with configurable hidden layer neurons. The default configuration is presented in table 2:

Table 2: Neural network architecture specifications.

Layer	Configuration	Function
Input Layer	Variable inputs	Data reception
Hidden Layer	2, 4, 6, 8, 10, 20, 30, 40, 50 neurons	Feature transformation
Output Layer	1 neuron	Prediction output

3.5. Performance Evaluation Metrics

The following metrics were used to evaluate neural network performance:

Mean Squared Error (MSE): The average squared difference between outputs and targets, where lower values indicate better performance and zero indicates no error as in Eq. (3).

$$MSE = \left(\frac{1}{n}\right)\Sigma(yi - \hat{y}i)^2 \quad (3)$$

Regression R Values: Measures the correlation between outputs and targets, where an R value of 1 indicates a close relationship and 0 indicates a random relationship.

Validation Performance: The best validation error achieved during training, recorded at the corresponding epoch number.

Error Histogram: Distribution analysis of prediction errors, where ideal performance shows tight clustering around zero with minimal spread.

3.6. Experimental Procedure

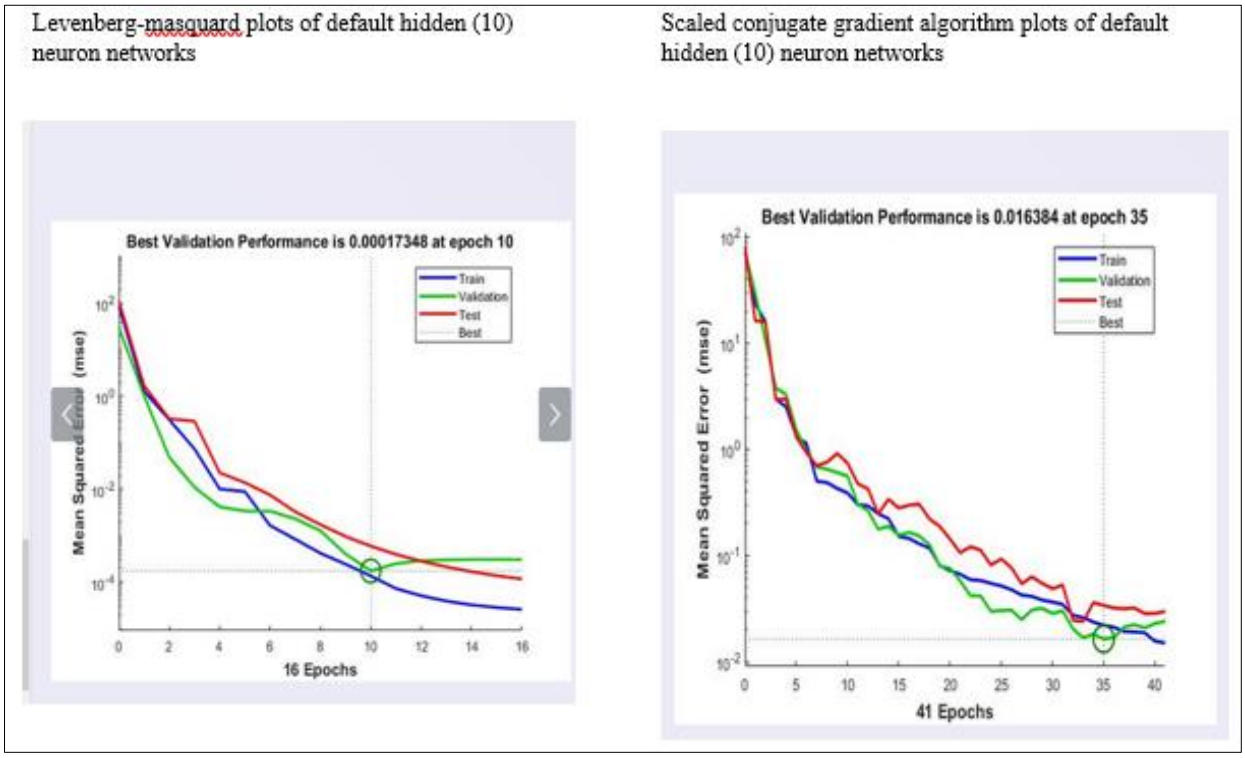
The experimental procedure followed these systematic steps:

1. Configure the neural network with specified hidden neuron count
2. Set training proportion and data division parameters
3. Select training algorithm (LM or SCG)
4. Execute training until validation-based stopping criteria
5. Record performance plots: validation performance, training state, error histogram, regression, and error fit
6. Repeat for all combinations of training proportions and hidden neuron configurations
7. Perform comparative analysis of results

4. Results and Discussions

4.1. Default Hidden Neuron Network Performance (10 Neurons, 70%)

The baseline configuration with 10 hidden neurons and 70% training proportion serves as the reference point for comparative analysis. Fig. 3 presents the validation performance plots for both the Levenberg-Marquardt and Scaled Conjugate Gradient algorithms.

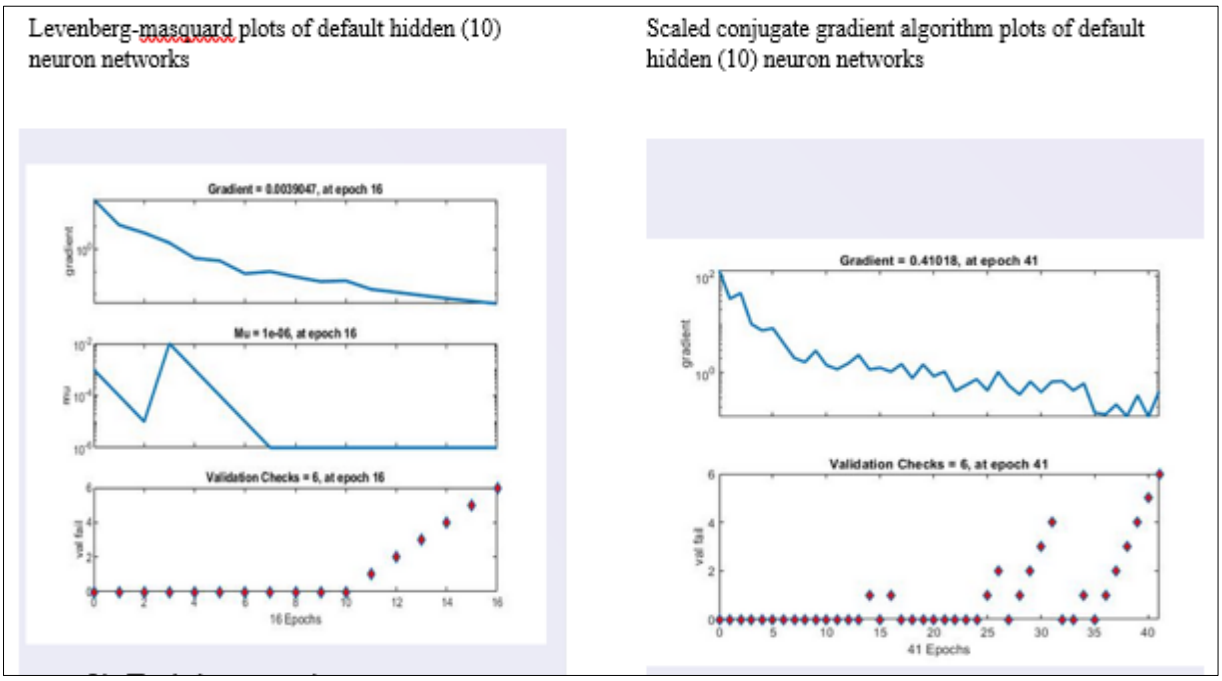


a: Performance by L-M (10) **b: Performance by SCG (10)**

Fig 3: Performance comparison of LM and SCG algorithms with 10 hidden neurons

Figure 3. is the plot of validation performance comparison between (a) Levenberg-Marquardt algorithm achieving best validation performance of 0.00017348 at epoch 10, and (b) Scaled Conjugate Gradient algorithm achieving best validation performance of 0.016384 at epoch 35, both using default 10 hidden neurons with 70% training proportion.

The results demonstrate that the Levenberg-Marquardt algorithm achieves significantly better validation performance (0.00017348) compared to the Scaled Conjugate Gradient (0.016384), representing an approximately 94-fold improvement. Additionally, the LM algorithm converges faster, reaching optimal performance at epoch 10 compared to epoch 35 for SCG.



a: Training state by L-M (10) **b: Training state by SCG (10)**

Fig 4: Training state by L-M and SCG with 10 hidden neurons

Figure 4 shows the training state comparison showing gradient, Mu (LM only), and validation checks for (a) Levenberg-Marquardt with gradient = 0.0039047 at epoch 16, Mu = 1e-06, and (b) Scaled Conjugate Gradient with gradient = 0.41018 at epoch 41.

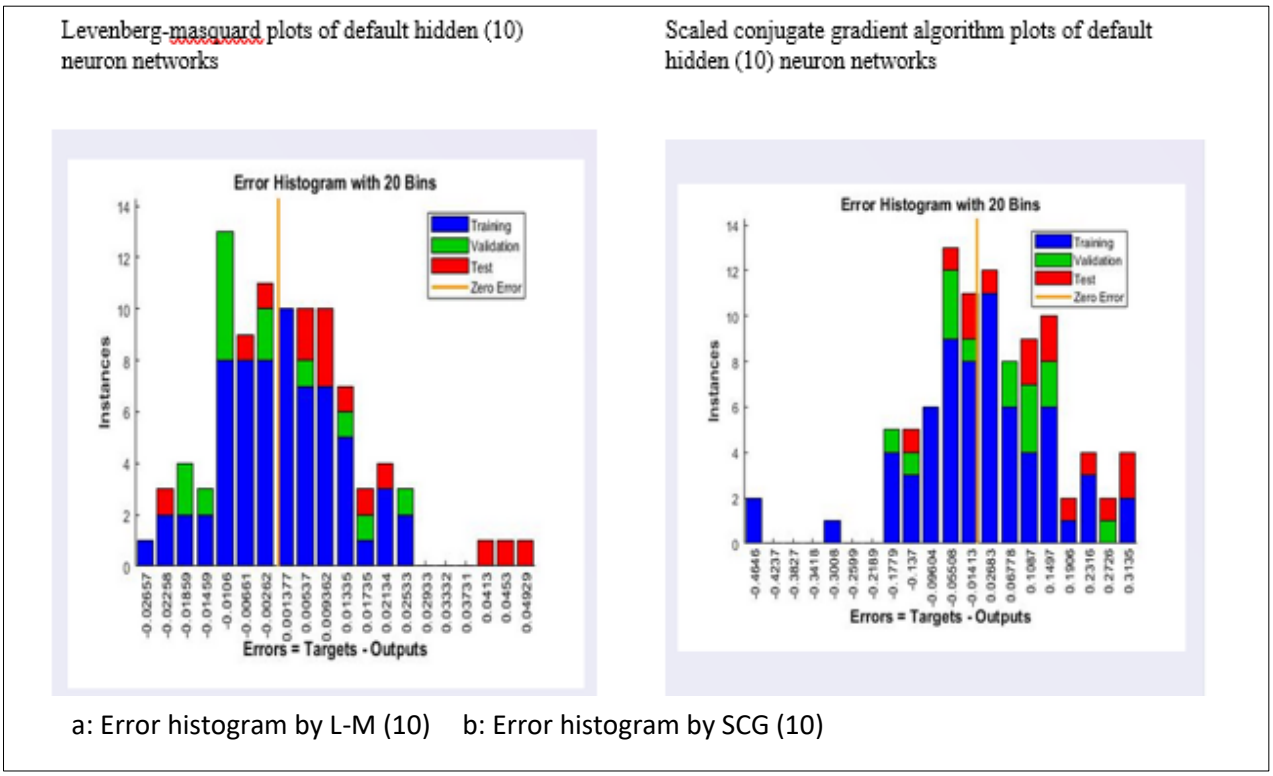


Fig 5: Error histogram comparison of LM and SCG algorithms

Error histogram with 20 bins comparing (a) Levenberg-Marquardt and (b) Scaled Conjugate Gradient algorithms for default 10 hidden neurons is plotted in figure 5. The LM algorithm shows tighter clustering around zero error.

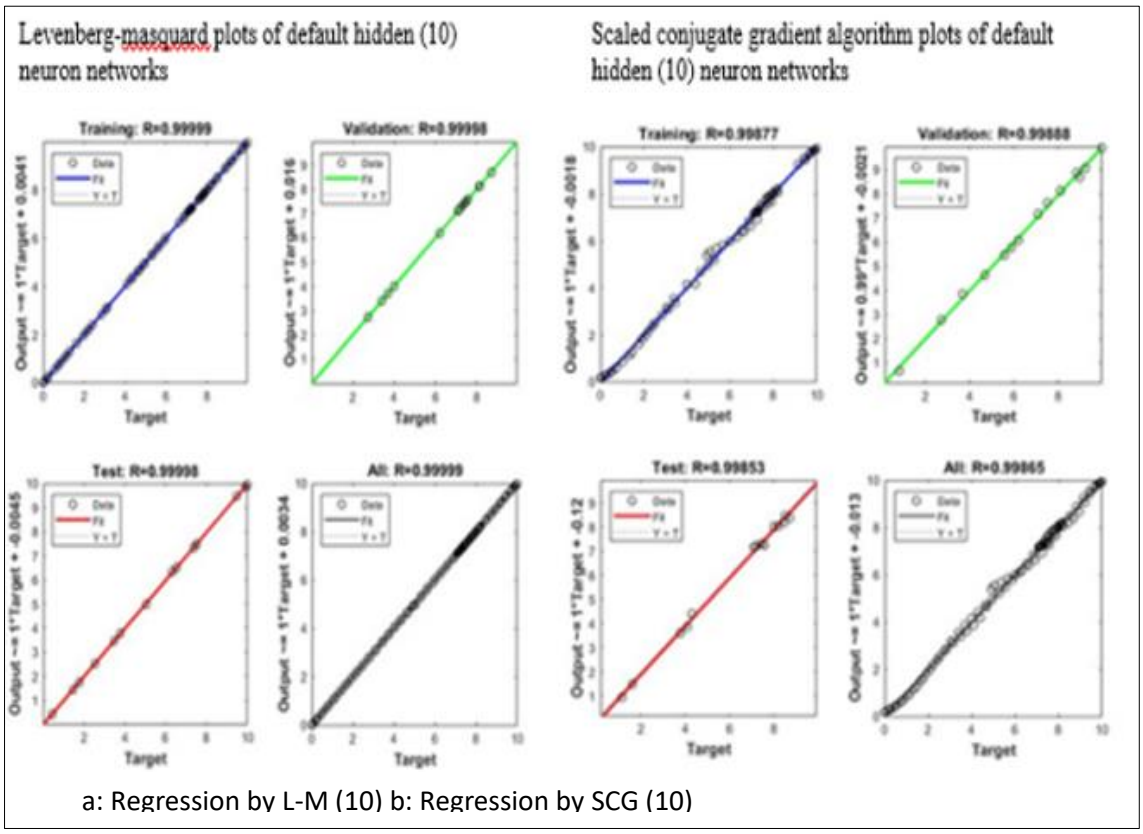
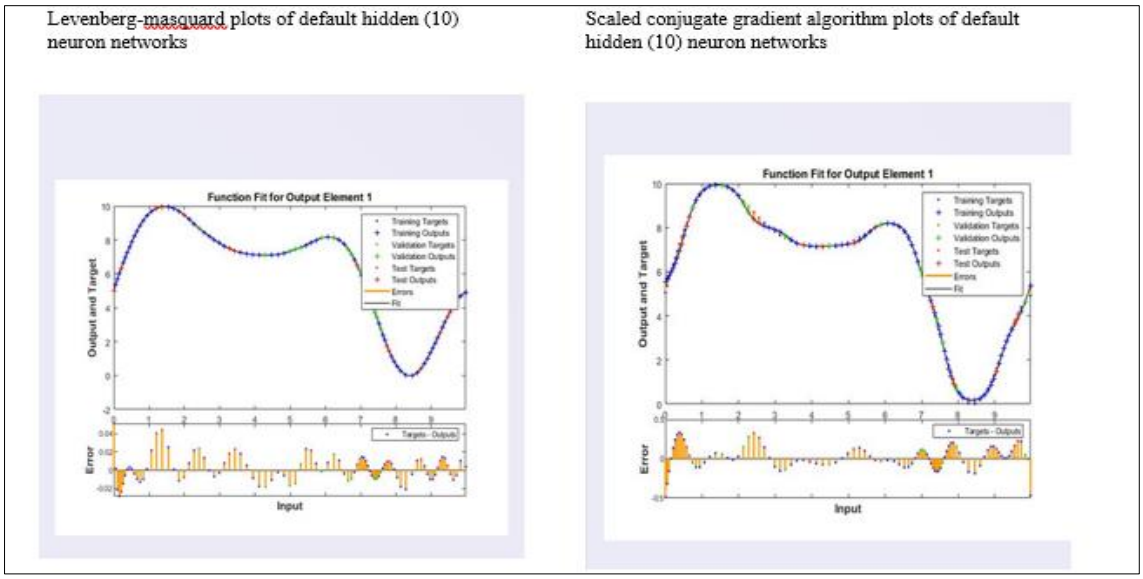


Fig 6: Regression analysis comparing LM and SCG algorithms

Figure 6 depicts the regression analysis showing R-values for training, validation, test, and overall performance: (a) Levenberg-Marquardt achieving R = 0.99999 for training and (b) Scaled Conjugate Gradient achieving R = 0.99877 for training indicating close relationship.



The function fit for output element 1 comparing (a) Levenberg-Marquardt and (b) Scaled Conjugate Gradient algorithms, showing training targets, validation targets, test targets, outputs, and errors is plotted in figure 7.

4.2. Hidden Neuron Variation Analysis

To evaluate the effect of hidden neuron count on input prediction accuracy, experiments were conducted with reduced (2 neurons) and increased (50 neurons) configurations. The result and observations were tabulated in table 3.

Table 3: Validation performance results for different hidden neuron configurations using Levenberg-Marquardt algorithm.

Hidden Neurons	Best Validation Performance	Epoch	Observation
2	1.1521	18	Underfitting - insufficient complexity
6	0.0033	30	Approaching optimal
8	0.00012831	94	Best performance
10 (default)	0.00017348	10	Good performance
20	0.0071551	8	Performance degradation begins
30	0.0072545	8	Overfitting tendency
40	Dispersed error	-	Underfitting due to complexity
50	Dispersed error	-	Severe underfitting

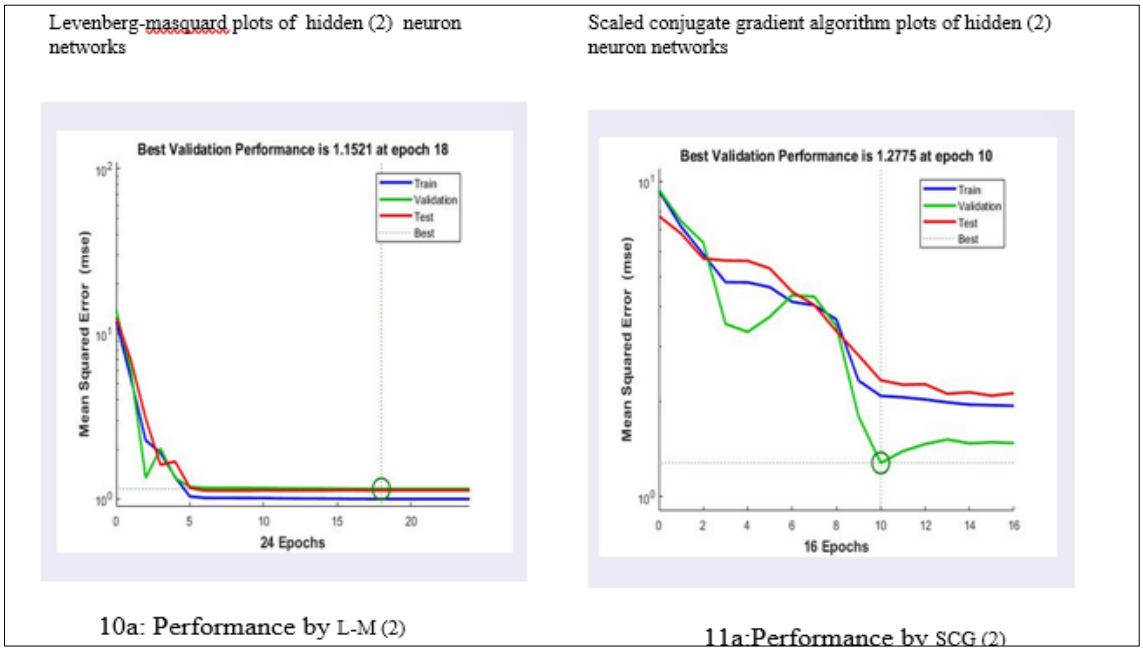


Fig 8: Performance by LM and SCG with 2 hidden neurons

Figure 8 shows the validation performance with 2 hidden neurons: (a) Levenberg-Marquardt achieving best performance of 1.1521 at epoch 18, and (b) Scaled Conjugate Gradient achieving 1.2775 at epoch 10, demonstrating underfitting with insufficient network complexity.

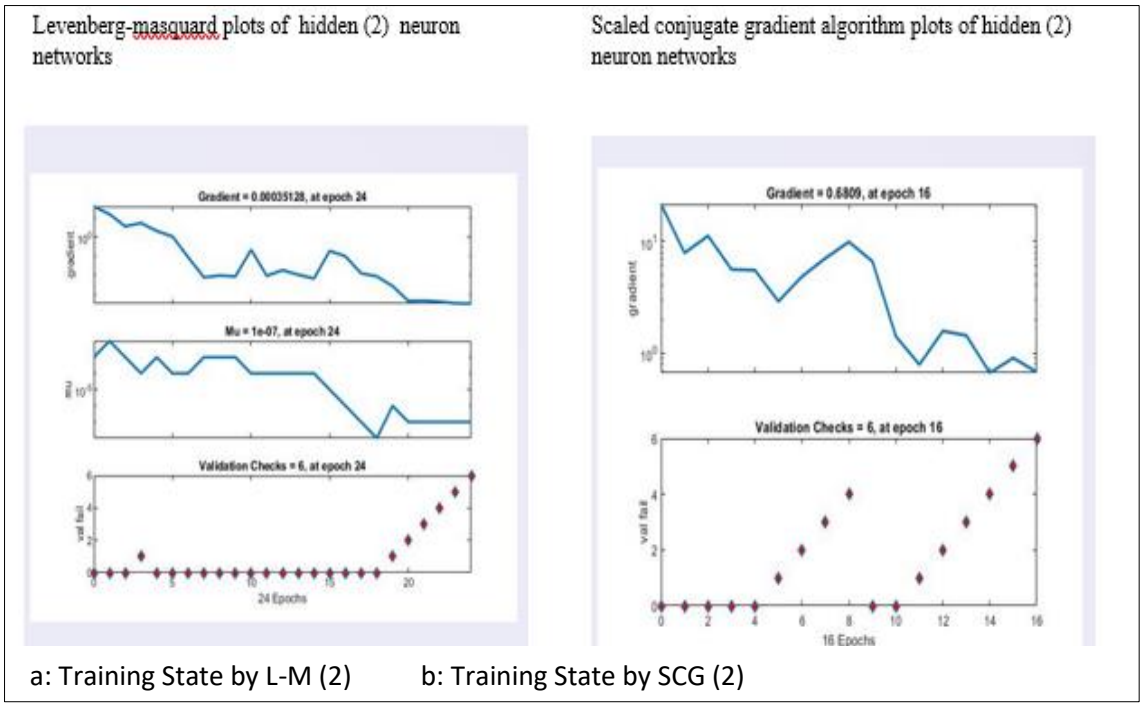


Fig 9: Training state by LM and SCG with 2 hidden neurons

Figure 9 shows the training state for 2 hidden neurons: (a) LM with gradient = 0.035123 at epoch 24, Mu = 1e-67, and (b) SCG with gradient = 0.6809 at epoch 16. The Levenberg-Marquardt model is characterized by rapid convergence, a greater decline in error early on, a lower mean squared error, a higher computation for each epoch, and a sensitivity to the initial weight. There is less sensitivity to weight and lesser computation per epoch, a slower convergence as seen by the scale conjugate gradient, a gradual fall in error as shown by the error reduction plot, and a mean squared error that is larger than LM.

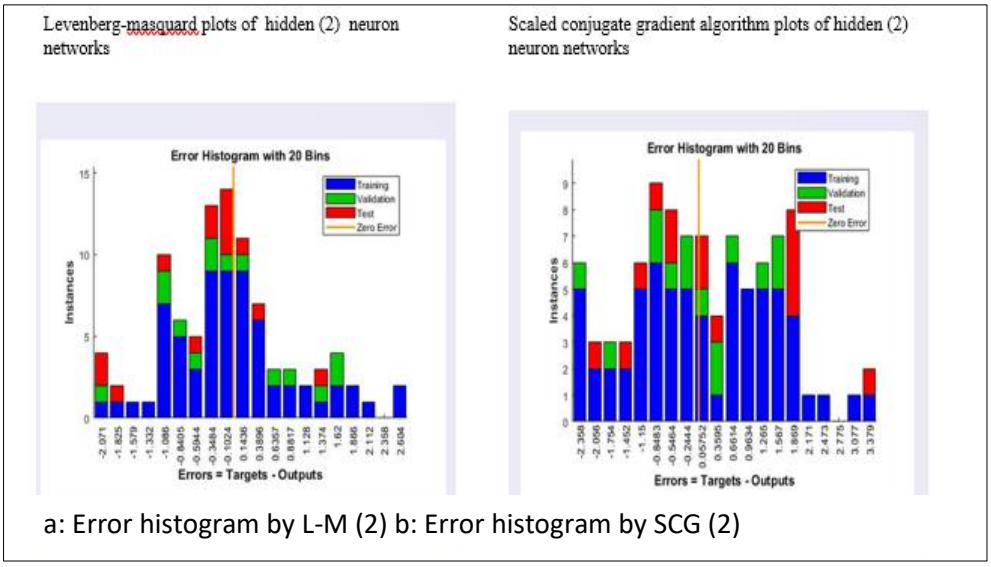


Fig 10: Error histogram by LM and SCG with 2 hidden neurons

Error histogram comparison with 2 hidden neurons showing wider error distribution indicative of underfitting: (a) Levenberg-Marquardt and (b) Scaled Conjugate Gradient algorithms. The error histogram shows how the distribution of errors (the discrepancy between the expected and actual results) is distributed throughout the test and validation samples as shown in figure 10. Compared to the scaled conjugate gradient, which displays a wider range of values and more samples with larger errors, the Liebenberg marquared shows a high concentration around zero error, indicating a strong match.

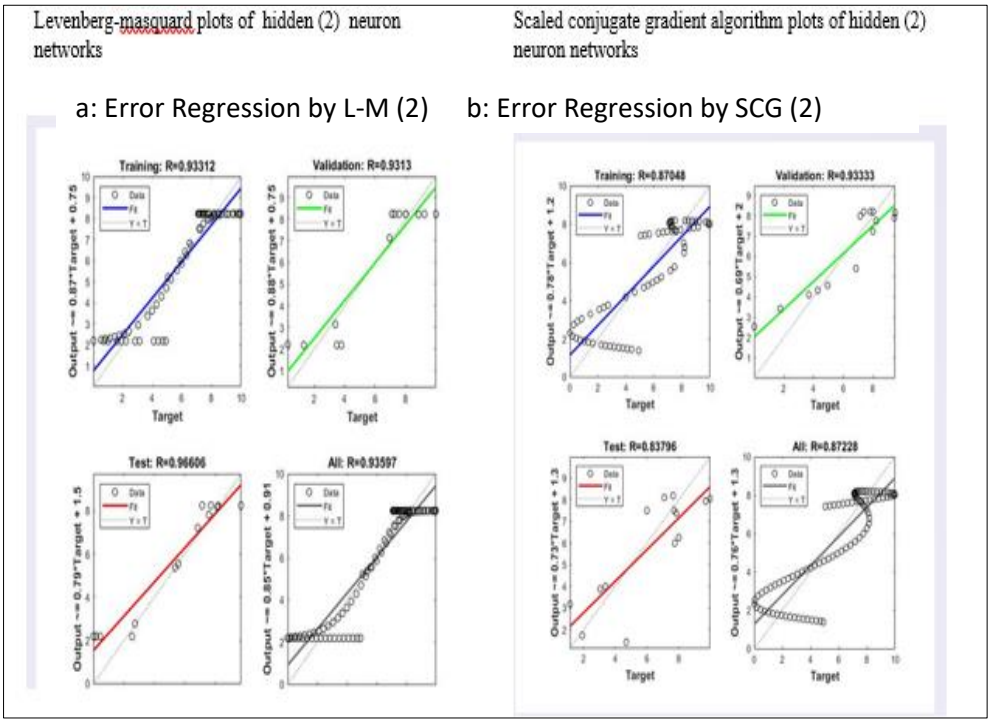


Fig 11: Regression analysis by LM and SCG with 2 hidden neurons

Figure 11 illustrates the regression analysis with 2 hidden neurons: (a) LM achieving R = 0.93312 for training, R = 0.93130 for validation, R = 0.96606 for test, and R = 0.93597 overall; (b) SCG achieving R = 0.87048 for training, R = 0.83333 for validation. Using a two-neurons-hidden network, while scale conjugate gradient achieves faster and lower error

reduction, Levenbeng marquardt outperforms it in terms of error regression, but it uses more compute and memory. For bigger models, where LM isn't feasible, SSG is the better option.

Fig 12: Error training fit by LM and SCG with 2 hidden neurons

Function fit was illustrated in figure 12 with 2 hidden neurons: (a) Levenberg-Marquardt showing tighter fit and (b) Scaled Conjugate Gradient showing higher training error fit indicative of potential underfitting. Scale conjugate gradient shows a higher training error fit, it may underfit if convergence stalls, slow convergence, more epochs to decent fit, and low capacity to sensivity to complexity. On the other hand, the final training fit error is lower, tighter, better fit to training data, faster, smoother, and more accurate.

4.3. Training Proportion Analysis

Comprehensive analysis of different training proportions reveals significant variations in validation performance and error minimization outcomes.

Table 4: Validation performance comparison across different training proportions for both algorithms with 10 hidden neurons.

Training %	LM Best Performance	LM Epoch	SCG Best Performance	SCG Epoch
80%	8.194×10-7	363	Higher error	-
70% (default)	0.00017348	10	0.016384	35
60%	1.910×10-5	512	Better than 70%	-
50%	Moderate	-	Moderate	-
40%	Higher error	-	Higher error	-
30%	0.04494	113	0.8044598	11

4.4. Error Histogram Analysis by Training Proportion

Table 5: Error histogram ranking by training proportion for Levenberg-Marquardt algorithm (1 = least error dispersion, 5 = most error dispersion).

Training %	LM Error Ranking	SCG Error Ranking	Overall Assessment
60%	1 (Least)	1 (Least)	Optimal for error histogram
80%	2	2	Second best
50%	3	3	Moderate
40%	4	4	Below optimal
70%	5 (Most)	5 (Most)	Highest error dispersion

From table 5, the error histogram analysis reveals that 60% training proportion produces the least error dispersion for both algorithms, suggesting this as the optimal configuration for error histogram minimization. Notably, the default 70% proportion shows the highest error dispersion, indicating that the commonly used default may not be optimal for all applications.

4.5. Discussion

4.5.1. Input Neuron Prediction Based on Error Minimization

The primary objective of this research—prediction of input neurons based on the outcome of error minimization of percentage proportion of hidden neurons—yields several significant findings.

The comprehensive analysis demonstrates that the Levenberg-Marquardt algorithm consistently outperforms the Scaled Conjugate Gradient algorithm across all evaluation metrics, establishing a clear preference for LM in input neuron prediction applications.

The validation performance analysis reveals a hierarchical ranking of training proportions for optimal input prediction: 80% achieves the best validation performance with an error value of 8.194×10^{-7} at epoch 363, followed by 60% with 1.910×10^{-5} at epoch 512, and finally 70% with 0.00017348 at epoch 10. This ranking contradicts the conventional wisdom of using 70% as the default training proportion and suggests that higher training proportions may be more suitable for applications requiring maximum validation performance.

4.5.2. Optimal Hidden Neuron Configuration

The hidden neuron analysis provides critical insights for input neuron prediction. The configuration with 8 hidden neurons produces the best validation performance with minimal error of 0.00012831 at epoch 94, closely approaching the optimal default target of 10 neurons. This finding suggests that for the input neuron prediction task, slightly fewer hidden neurons than the default may yield superior results.

Configurations with 2-6 hidden neurons exhibit underfitting characteristics, with validation errors ranging from 1.1521 (2 neurons) to 0.0033 (6 neurons). Conversely, increasing the number of neurons beyond 20 leads to performance degradation, with 40 and 50 neurons showing dispersed error histograms characteristic of underfitting due to excessive network complexity without sufficient training data to support the increased parameter space.

4.5.3. Error Histogram Implications

A particularly noteworthy finding is that 60% training proportion produces the least error dispersion in the histogram for both LM and SCG algorithms. This suggests that for applications where error distribution consistency is critical—such as medical diagnostics or safety-critical industrial systems—a 60% training proportion may be optimal despite not achieving the absolute lowest validation error.

The error histogram ranking (60% → 80% → 50% → 40% → 70%) for the Levenberg-Marquardt algorithm indicates that the commonly used 70% default proportion actually exhibits the highest error dispersion. This counterintuitive result highlights the importance of empirical validation rather than reliance on default parameters in neural network applications.

4.5.4. Comparative Algorithm Performance

The comparative analysis between Levenberg-Marquardt and Scaled Conjugate Gradient algorithms reveals consistent superiority of LM across multiple evaluation criteria as summarized in table 6.

Table 6: Comparative summary of Levenberg-Marquardt versus Scaled Conjugate Gradient algorithm performance characteristics.

Criterion	Levenberg-Marquardt	Scaled Conjugate Gradient	Advantage
Validation Performance (10 neurons, 70%)	0.00017348	0.016384	LM (94× better)
Convergence Speed (epochs)	10	35	LM (3.5× faster)
Training R-value	0.99999	0.99877	LM
Error Concentration	Tight clustering	Wider spread	LM
Memory Requirements	Higher	Lower	SCG
Computational Efficiency	More per epoch	Less per epoch	Context-dependent

While the Scaled Conjugate Gradient algorithm offers advantages in memory efficiency and lower computational cost per epoch, these benefits are outweighed by the superior prediction accuracy and faster overall convergence of the Levenberg-Marquardt algorithm for input neuron prediction tasks. The LM algorithm's ability to achieve approximately 94-fold better validation performance makes it the recommended choice for applications where prediction accuracy is the primary concern.

4.5.5. Practical Implications

The findings of this study have significant practical implications for various application domains. In medical diagnostics, where error minimization is critical for patient safety, the recommendation to use the Levenberg-Marquardt algorithm with 80% training proportion and 8 hidden neurons provides a concrete configuration guideline. For industrial automation and fault detection applications, the 60% training proportion offers the best error distribution consistency.

The study also reveals that default parameters commonly used in neural network toolboxes may not be optimal for specific prediction tasks. Practitioners should conduct systematic evaluations of training proportions and hidden neuron configurations rather than relying on default values, particularly for applications where prediction uncertainty has significant consequences.

5. Conclusion

This study has systematically investigated input neuron prediction based on the error minimization outcomes of percentage proportion of hidden neurons using both the Levenberg-Marquardt and Scaled Conjugate Gradient algorithms. The comprehensive analysis across multiple training proportions (80%, 70%, 60%, 50%, 40%, and 30%) and hidden neuron configurations (2, 4, 6, 8, 10, 20, 30, 40, and 50) has yielded several key findings that contribute to the field of neural network optimization and uncertainty minimization.

The primary conclusions of this research are:

- **Algorithm Superiority:** The Levenberg-Marquardt algorithm consistently outperforms the Scaled Conjugate Gradient algorithm for input neuron prediction, achieving approximately 94-fold better validation performance (0.00017348 vs. 0.016384) with 3.5 times faster convergence at the default configuration.
- **Optimal Training Proportion for Validation Performance:** Given 10 input hidden neurons, the optimal training proportion for validation performance is 80%, achieving an error of 8.194×10-7 at epoch 363, followed by 60% (1.910×10-5 at epoch 512), and then 70% (0.00017348 at epoch 10).
- **Optimal Training Proportion for Error Histogram:**

A 60% training proportion produces the least error dispersion in the histogram for both algorithms, establishing this as the optimal configuration for applications requiring consistent error distribution.

- **Optimal Hidden Neuron Configuration:** For the input neuron prediction task, 8 hidden neurons produce the best validation performance with minimal error of 0.00012831 at epoch 94. Configurations with 40-50 neurons exhibit underfitting characteristics due to excessive network complexity.
- **Default Parameter Reconsideration:** The commonly used 70% default training proportion shows the highest error dispersion in the histogram analysis, suggesting that practitioners should conduct empirical validation rather than relying on default parameters.

With identical 60% proportions of 10 hidden neurons trained using both approaches, a superior function fit is attained with a minimal error of 0.01, consistent with the validation performance and error histogram findings. These results provide concrete guidelines for neural network practitioners in selecting optimal configurations for input neuron prediction tasks.

The practical applications of these findings extend to medical image processing, medical diagnostics (illness classification from symptoms or images), industrial automation, route and traffic prediction, vehicle and fault detection, and defect detection in manufacturing. The established superiority of the Levenberg-Marquardt algorithm and the identified optimal configurations provide a foundation for improved prediction accuracy and reduced uncertainty in these critical application domains.

Author Contributions

Onah Okechukwu Thomas: Conceptualization, Investigation, Methodology, Resources, Supervision, Writing – original draft; Nebo Ephraim Ugochukwu: Software, Data curation, Formal analysis, Visualization; Aka Christian Chikezie: Software, Visualization, Validation, Writing – review & editing

Funding

This research did not receive external funding from any agencies.

Ethical statement

Not applicable

Data availability statement

Data are included in the article.

Conflict of Interest

The authors declare no conflict of interest.

References

1. Chen M, Challita U, Saad W, Yin C, Debbah M. Artificial neural networks-based machine learning for wireless networks: a tutorial. IEEE Commun Surv Tutor. 2019;21(4):3039-71. doi:10.1109/comst.2019.2926625
2. van Gerven M, Bohte S. Editorial: artificial neural networks as models of neural information processing. Front Comput Neurosci. 2017;11:114. doi:10.3389/fncom.2017.00114
3. Kostryzhev A, Singh N, Chen L, Killmore C, Pereloma E. Comparative effect of Mo and Cr on microstructure and mechanical properties in NbV-microalloyed bainitic steels. Metals. 2018;8(2):134. doi:10.3390/met8020134
4. Li H, Zhang Z, Liu Z. Application of artificial neural networks for catalysis: a review. Catalysts. 2017;7(10):306. doi:10.3390/catal7100306
5. Elsheikh AH, Sharshir SW, Abd Elaziz M, Kabeel AE, Guilan W, Haiou Z. Modeling of solar energy systems using artificial neural network: a comprehensive review. Sol Energy. 2019;180:622-39. doi:10.1016/j.solener.2019.01.037
6. Sultana M, Arshad U, Akgül A, Khalid M. Numerical solutions of fractional systems using Bessel artificial neural network based integrated intelligent computing. J Comput Nonlinear Dyn. 2025;20(5):1-16. doi:10.1115/1.4068236
7. Chaudhari KG. Analysis on transport layer protocols and its developments. SSRN Electron J. 2018. doi:10.2139/ssrn.3729064
8. Rakhmandasari A, Mahmudy WF, Yulianti T. Kenaf plant pest and disease detection using faster regional based convolutional neural network. Indones J Electr Eng Comput Sci. 2021;24(1):198-207. doi:10.11591/ijeecs.v24.i1.pp198-207
9. Lin YC, Wu KD, Shih WC, Hsu PK, Hung JP. Prediction of surface roughness based on cutting parameters and machining vibration in end milling using regression method and artificial neural network. Appl Sci (Basel). 2020;10(11):3941. doi:10.3390/app10113941
10. Lalwani V, Sharma P, Pruncu CI, Unune DR. Response surface methodology and artificial neural network-based models for predicting performance of wire electrical discharge machining of Inconel 718 alloy. J Manuf Mater Process. 2020;4(2):44. doi:10.3390/jmmp4020044
11. David Lide R. Methods of conjugate gradients for solving linear systems. In: A century of excellence in measurements, standards, and technology. Boca Raton (FL): CRC Press; 2018. p. 81-5. doi:10.1201/9781351069397-24
12. Møller MF. A scaled conjugate gradient algorithm for fast supervised learning. Neural Netw. 1993;6(4):525-33. doi:10.1016/s0893-6080(05)80056-5
13. Chukwukwe E, Palama V, Emmanuel EJ, Ogbonna CU. Phytoremediation potential of indigenous plants in Abia State, southeastern Nigeria: a multi-site and multi-season assessment. Int J Curr Microbiol App Sci. 2025;14(9):97–112. doi:10.20546/ijcmas.2025.1409.011.
14. Levenberg K. A method for the solution of certain non-linear problems in least squares. Quart Appl Math. 1944;2(2):164-8. doi:10.1090/qam/10666
15. Marquardt DW. An algorithm for least-squares estimation of nonlinear parameters. J Soc Ind Appl Math. 1963;11(2):431-41. doi:10.1137/0111030
16. Hagan MT, Menhaj MB. Training feedforward networks with the Marquardt algorithm. IEEE Trans Neural Netw. 1994;5(6):989-93. doi:10.1109/72.329697
17. Vlacic L. Learning and soft computing, support vector machines, neural networks, and fuzzy logic models, Vojislav Kecman; MIT Press, Cambridge, MA, 2001, ISBN 0-262-11255-8, 2001, pp. 578. Neurocomputing. 2002;47(1-4):305-7. doi:10.1016/s0925-2312(01)00685-3

18. Adams MF. Multigrid equation solvers for large scale nonlinear finite element simulations [dissertation]. Fort Belvoir (VA): Defense Technical Information Center; 1999.
19. Ferjaoui R, Cherni MA, Abidi F, Zidi A. Deep residual learning based on ResNet50 for COVID-19 recognition in lung CT images. In: Proceedings of the 2022 8th International Conference on Control, Decision and Information Technologies (CoDIT); 2022 May 17-20; Istanbul, Turkey. Piscataway (NJ): IEEE; 2022.
20. Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, *et al*. Going deeper with convolutions. In: Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2015 Jun 7-12; Boston, MA. Piscataway (NJ): IEEE; 2015.
21. Chukwukwe EU, Deigh C, Nwaogwugwu CJ, Nwankwo JC, Chukwuemeka US. Eco-friendly nanocomposites for the degradation of emerging contaminants in wastewater systems. Asian J Adv Res Rep. 2025;19(8):256–281. doi:10.9734/ajarr/2025/v19i81129
22. Bhatia S, Bhardwaj D, Akash A, Bhardwaj L. Prediction in stock market value using artificial intelligence. In: Proceedings of the 2024 IEEE 4th International Conference on ICT in Business Industry & Government (ICTBIG); 2024 Nov 8-9; Indore, India. Piscataway (NJ): IEEE; 2024. p. 1-4.
23. Soni VD. Role of AI in industry in emergency services. SSRN Electron J. 2018. Available from: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3691783
24. Bengio Y, Simard P, Frasconi P. Learning long-term dependencies with gradient descent is difficult. IEEE Trans Neural Netw. 1994;5(2):157-66. doi:10.1109/72.279181
25. Luo P, Wang X, Shao W, Peng Z. Towards understanding regularization in batch normalization. arXiv:1809.00846. 2018. Available from: http://arxiv.org/abs/1809.00846

How to Cite This Article

Ugochukwu NE, Thomas OO, Chikezie AC. Input neuron prediction based on error minimization of percentage proportion of hidden neurons: a comparative study of Levenberg–Marquardt and scaled conjugate gradient algorithms. Int J Artif Intell Eng Transform. 2025;6(2):179–190. doi:10.54660/IJAIET.2025.6.2.179-190.

Creative Commons (CC) License