Comparative Analysis of ANN-Based Solar Radiation Forecasting Using GUI-Based and MATLAB Command Approaches

Student name 1, s123456, Student name 2, s123456, Student name 3, s123456,

***Abstract*—This study evaluates the performance of artificial neural network (ANN) models for weather data analysis using datasets collected from multiple weather stations in Sydney, Australia. The research compares ANN models constructed using a GUI-based tool (nnstart) and MATLAB commands (newff, traingd, newrb) to model solar exposure patterns over a year, focusing on the accuracy and generalization of the models across different seasons.**

***Index Terms*—Artificial Neural Networks (ANN),GUI-Based Neural Network Tools,Levenberg-Marquardt Algorithm, FFBP (Feedforward Backpropagation),RBF (Radial Basis Function),GUI-Based Neural Network Tools,**

I. Introduction

WEATHER forecasting is vital for numerous applications, including agriculture, transportation, and disaster management. ANN models have emerged as effective tools for capturing complex weather patterns and improving prediction accuracy. This study aims to assess the effectiveness of ANN models in modeling daily weather data from various Sydney stations over four seasons. The study also explores the comparative advantages of using GUI-based tools versus MATLAB commands for ANN modeling in weather data analysis.

II. Methods

A. Data Collection and Preprocessing

Daily solar radiation data, along with meteorological variables, was collected from the following weather stations for the period from May 2023 to May 2024, resulting in a dataset of 1987 rows:

- Sydney City, Sydney NSW (Station No. 0067119)
- Blacktown, Horsley Park NSW (Station No. 066214)
- Earlwood, Canterbury Racecourse AWS NSW (Station No. 066194)
- Randwick, Sydney Airport AMO (Station No. 066037)
- Rozelle, Terry Hills AWS (Station No. 066059)

The collected data included the following meteorological variables as inputs:

- Minimum Temperature
- Maximum Temperature
- Rainfall
- Maximum Wind Speed

The output variable, solar exposure, was also included in the dataset. After collecting and cleaning the data, Min-Max Normalization was applied to normalize the input features within a specified range. The normalized dataset was then divided into training (70%), validation (15%), and testing (15%) sets to facilitate model evaluation.

B. GUI-Based Approach (nnstart)

The GUI-based tool nnstart was utilized to explore various input combinations, including maximum temperature, minimum temperature, rainfall, maximum wind speed, and solar exposure, to optimize the input-output mapping performance for solar radiation forecasting. The Levenberg-Marquardt backpropagation algorithm was employed to train the neural network models.

C. MATLAB-Based Approach

ANN models were implemented using MATLAB commands newff, traingd, and newrb. Training rules such as feedforward gradient descent backpropagation and radial basis function were applied. Performance metrics similar to the GUI-based approach were measured for comparison.

III. Results and Analysis

A. GUI-Based Approach (nnstart)

The Levenberg-Marquardt backpropagation algorithm was employed to train the neural network models. ANN models were constructed using nnstart with varying configurations of hidden neurons and training epochs. Performance metrics, including the number of hidden neurons, epochs, simulation time taken, gradient values, validation checks, and Mean Squared Error (MSE), were recorded for each model. This comprehensive evaluation aimed to determine the optimal configuration for achieving accurate and efficient solar radiation forecasting.

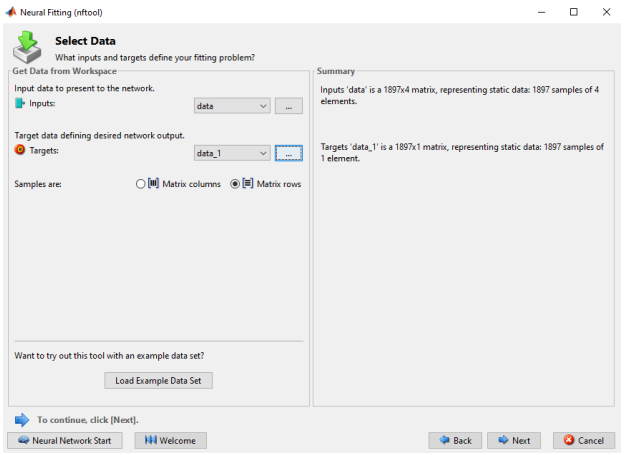

Fig. 1. Screenshot of Data Selection

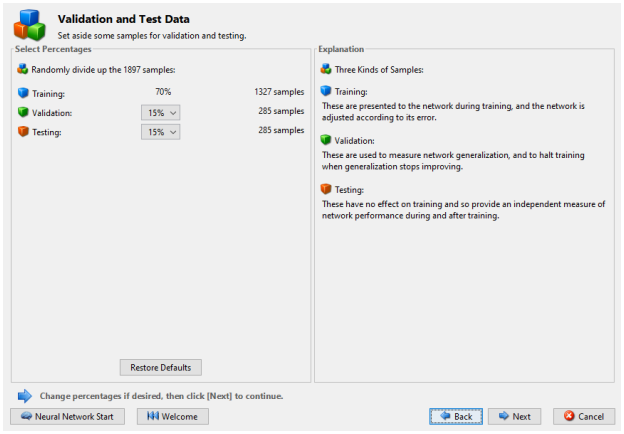

Fig. 2. Screenshot of Data Splitting

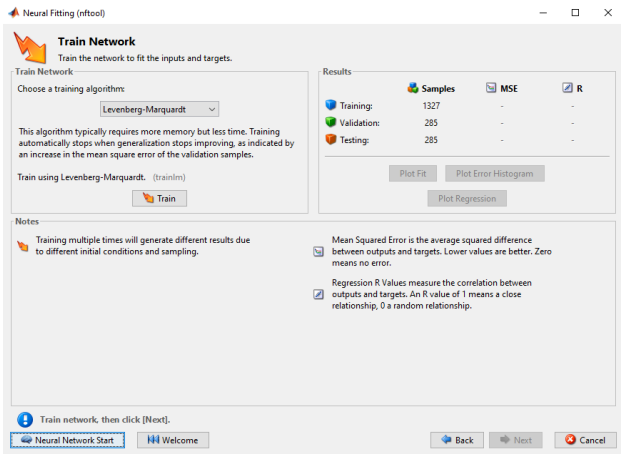

Fig. 3. Screenshot of Model Selection

1) Training with Different Numbers of Hidden Neurons: The GUI-based tool *nnstart* was utilized to explore various input combinations, including maximum temperature, minimum temperature, rainfall, maximum wind speed, and solar exposure, to optimize the input-output mapping performance for solar radiation forecasting. The training performance plot shows how the performance of the neural network improves over time (epochs) during the training process.

- Training Performance Plot: Shows how the neural network's performance improves over epochs during training. Includes training, validation, and test errors to evaluate the learning process.
- Fitting Plot: Illustrates how well the neural network model fits the training data by comparing predicted values to actual values.
- Regression Plot: Shows the relationship between actual outputs and network outputs, indicating the accuracy of the model's predictions.

Training with 10 Hidden Neurons

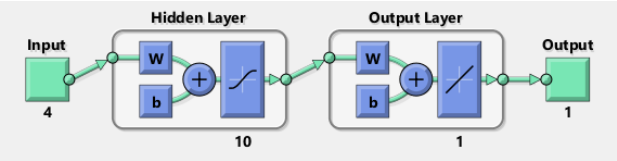

Fig. 4. 10 Hidden Neurons

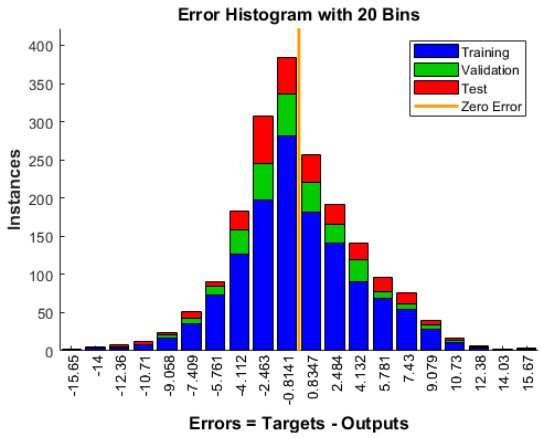

Fig. 5. Error Histogram

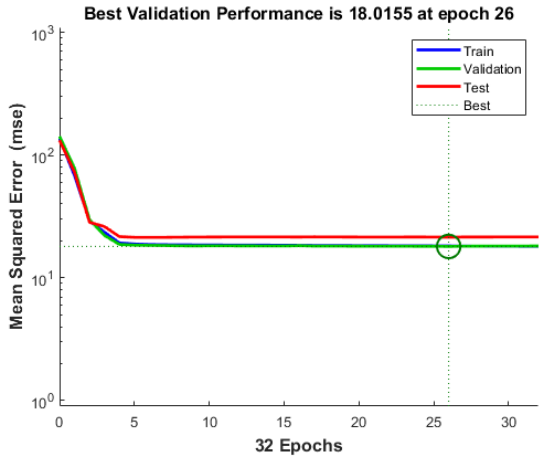

Fig. 6. Performance

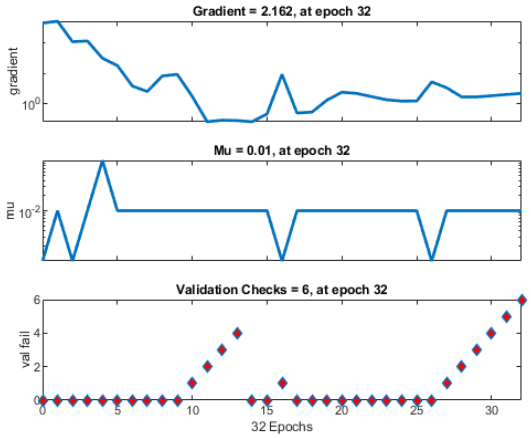

Fig. 7. Training State

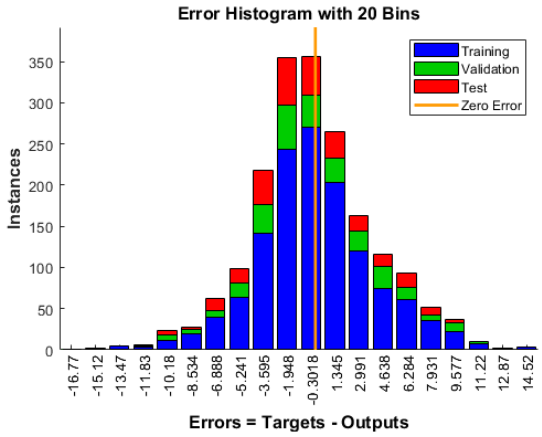

Fig. 10. Error Histogram

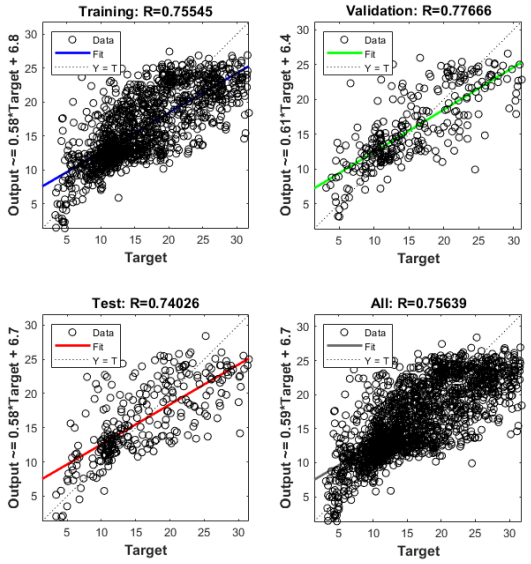

Fig. 8. Regression

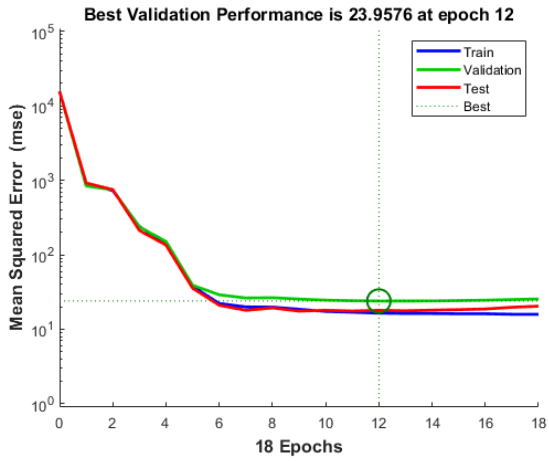

Fig. 11. Performance

Training with 50 Hidden Neurons

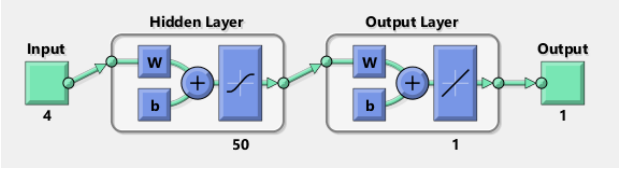

Fig. 9. 50 Hidden Neurons

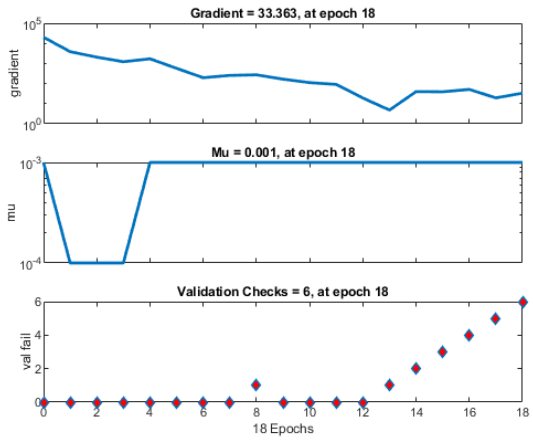

Fig. 12. Training State

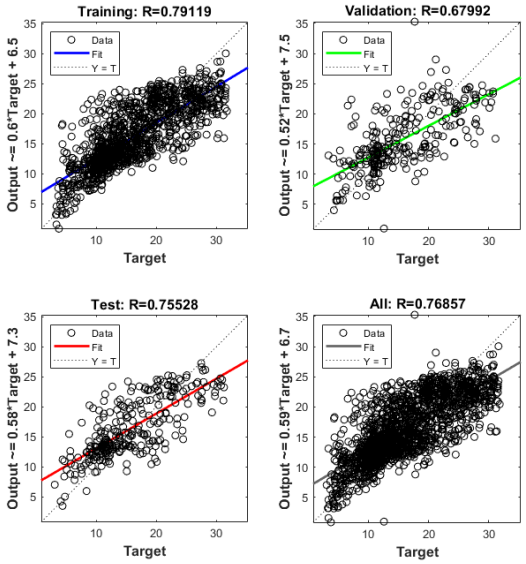

Fig. 13. Regression

Training with 100 Hidden Neurons Trained Model has shown as follows

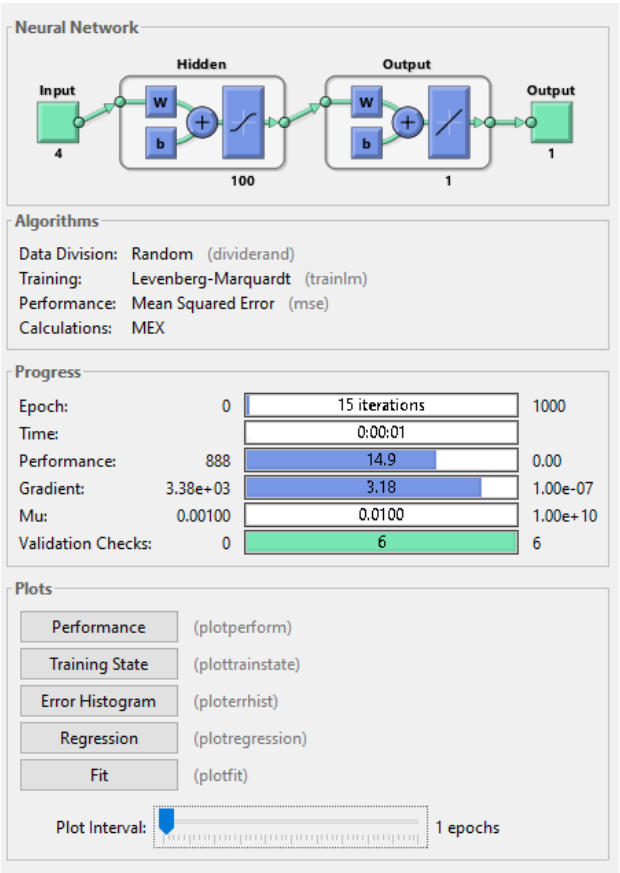

Fig. 14. 50 Hidden Neurons

Results

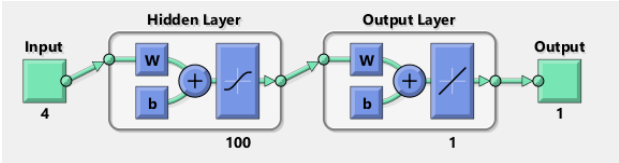

Fig. 15. 50 Hidden Neurons

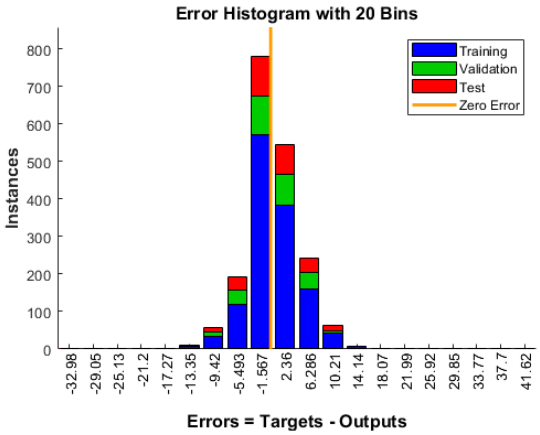

Fig. 16. Error Histogram

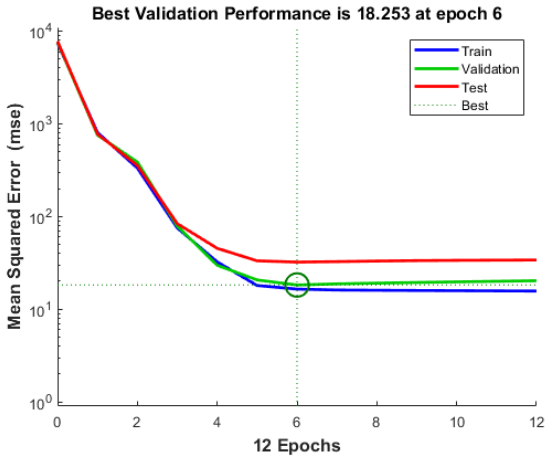

Fig. 17. Performance

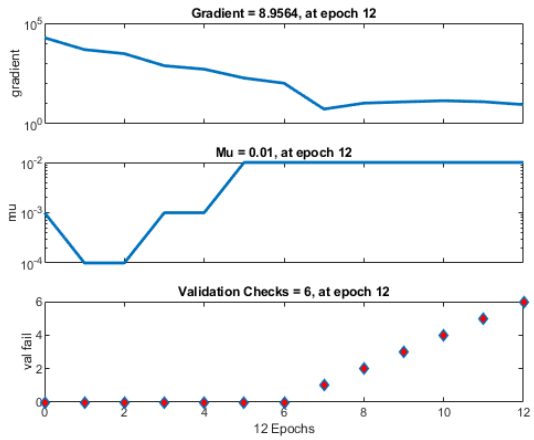

Fig. 18. Training State

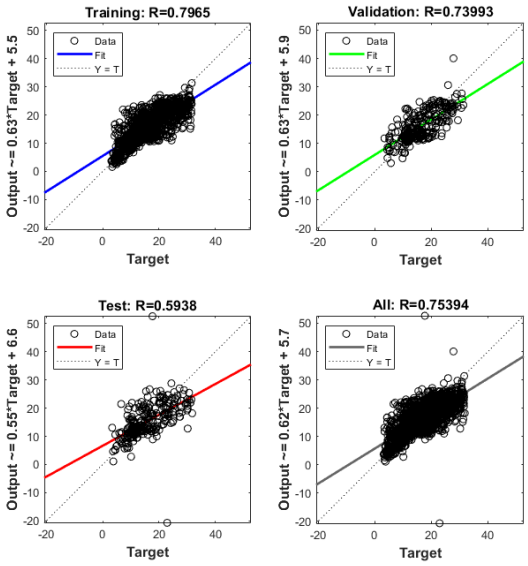

Fig. 19. Regression

TABLE I
MODEL TRAINING RESULTS

H.Neurons	Epochs	Sim.Time	Gradient	Val.Checks	MSE
10	32	0.00.00s	2.16	6	18.015
50	18	0.00.00s	33.4	6	23.95
100	12	0.00.01s	8.96	6	18.25

B. MATLAB Based Approach

Feed forward Gradient descent Back propagation Results

We utilized the same dataset for training our neural network using the Feedforward Gradient Descent Backpropagation algorithm. The script file `feed_forward.m` was developed and executed to train the model, employing the principles of backpropagation to adjust the network weights and biases iteratively for optimal performance.

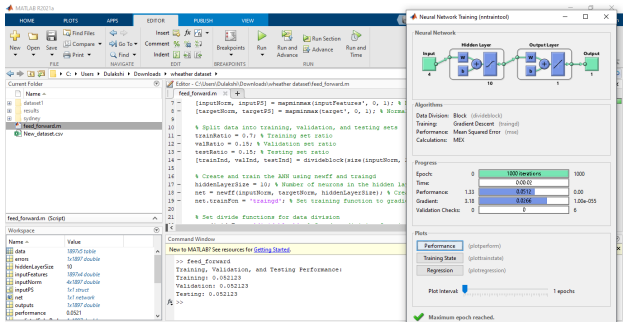

Fig. 20. Training

Generated Outputs

1) **Performance Plot:**
The performance plot illustrates the convergence of the training process over epochs. It provides insights into the model's learning dynamics, showcasing how the error or loss function changes with each iteration, indicating the network's ability to minimize errors.

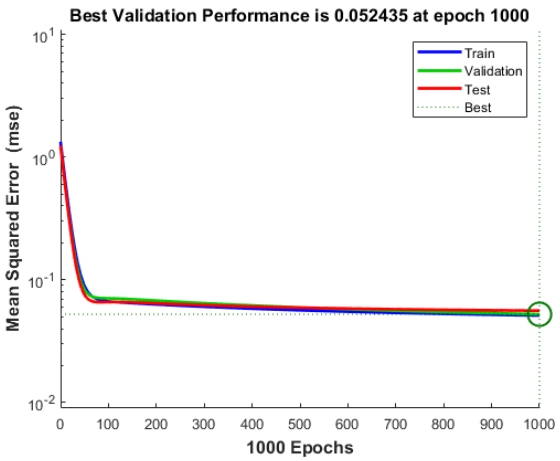

Fig. 21. Performance

2) **Regression Analysis:**
A regression analysis was conducted to evaluate the relationship between the predicted outputs and the actual target values. This analysis helps assess the accuracy and reliability of the neural network model in predicting outcomes based on input data.

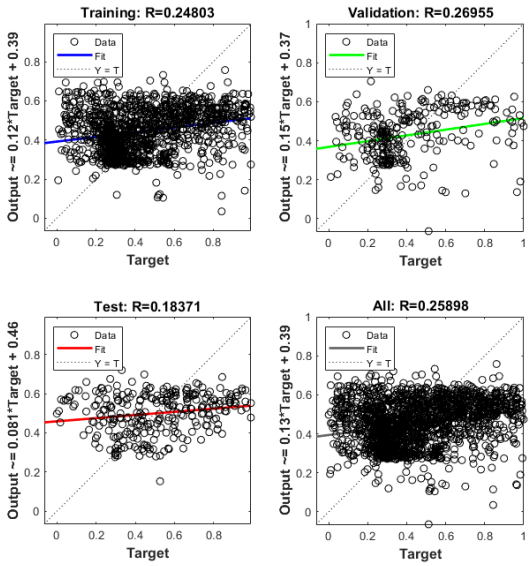

Fig. 22. Regression

3) **Training State:**
The training state of the neural network refers to its configuration and parameters after training. This includes the final weights and biases, as well as other network characteristics such as activation functions and learning rate settings.

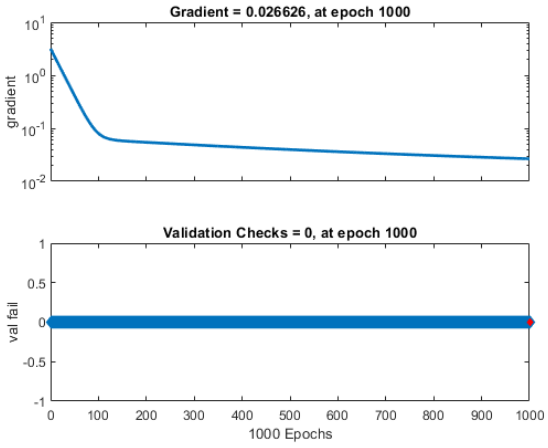

Fig. 23. Training State

4) **Actual vs. Predicted Results Plot:**
The actual vs. predicted results plot visually compares the model's predictions against the true values from the dataset. This plot provides a clear depiction of the model's performance in predicting outcomes, highlighting any discrepancies or patterns in prediction accuracy.

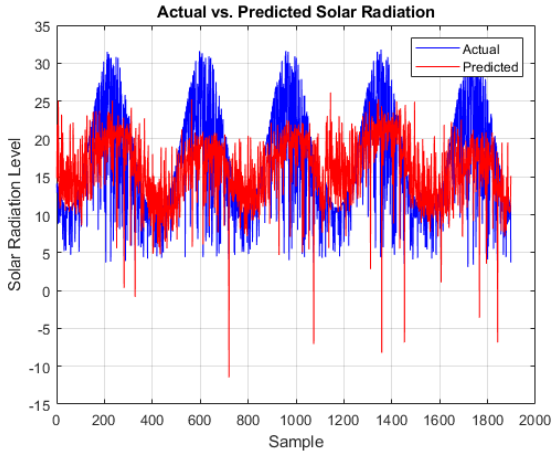

Fig. 24. Actual VS Predicted Results

Radial Basis Function (RBF) algorithm
We employed the same dataset for training our neural network using the Radial Basis Function (RBF) algorithm. The script file `radial_basis.m` was created and executed to train the model, utilizing the principles of RBF networks for pattern recognition and regression tasks.

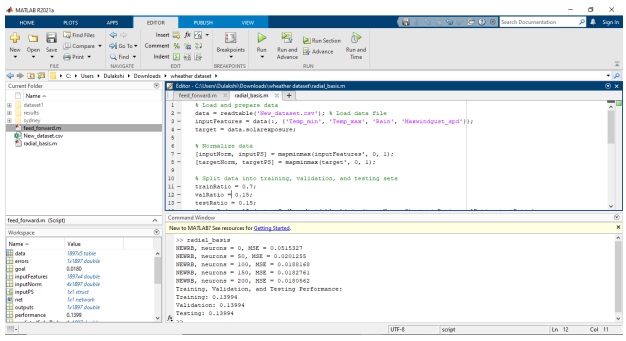

Fig. 25. Training

TABLE II
CHANGES OF NEURONS VS MSE

Neurons	MSE
0	0.0515
50	0.0201
100	0.0188
150	0.0182
200	0.0180

The training process involved adjusting the weights and biases of the network to minimize the Mean Squared Error (MSE) between the predicted outputs and the actual target values. Our target MSE value for this training was set to 0.18, indicating our desired level of accuracy in predicting outcomes. **Generated Outputs**

1) **Performance Plot:**
The performance plot illustrates the convergence of the training process over epochs. It shows how the MSE changes with each iteration, demonstrating the network's learning progress and its ability to approximate the target function.

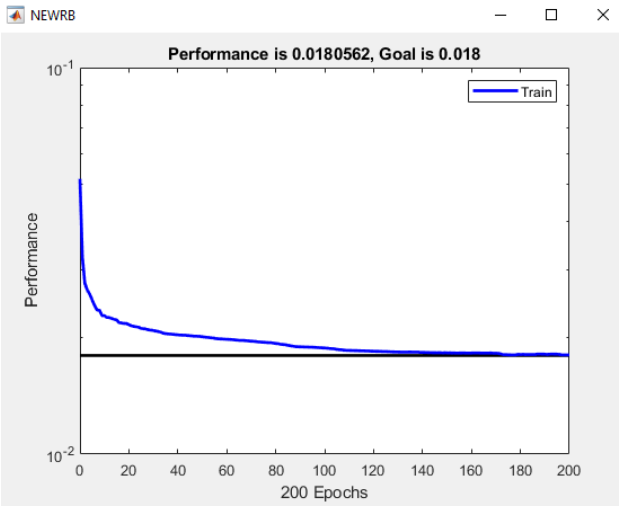

Fig. 26. Performance

2) **Regression Analysis:**
A regression analysis was conducted to evaluate the relationship between the predicted outputs and the actual target values. This analysis helps assess the accuracy and reliability of the neural network model in predicting outcomes based on input data.

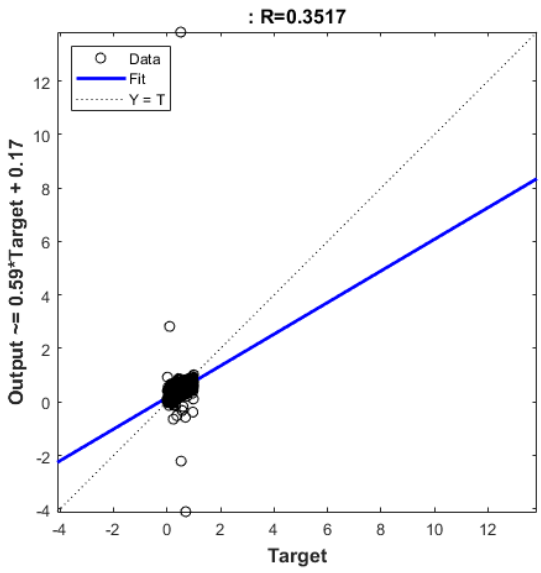

Fig. 27. Regression

A regression analysis was conducted to evaluate the predictive accuracy of the RBF neural network. We compared the predicted outputs against the actual target values to assess the model's ability to generalize and make accurate predictions on unseen data.

IV. Comparison of Performance

In this section, we compare the performance of the artificial neural network (ANN) models developed using GUI-based and MATLAB command approaches for solar radiation forecasting.

TABLE III
Training, Validation, Testing performance of Algorithms

Algorithm	Training	Validation	Testing
FFBP	0.0521	0.0521	0.052
RBF	0.1399	0.139	0.139

A. GUI-Based Approach (nnstart)

The GUI-based approach utilized the nnstart tool to construct ANN models with varying configurations, including different numbers of hidden neurons and training epochs. The Levenberg-Marquardt backpropagation algorithm was employed for training the models.

The results show that the model with 100 hidden neurons achieved the lowest Mean Squared Error (MSE) of 18.25, indicating better accuracy in solar radiation forecasting compared to the models with 10 and 50 hidden neurons.

B. MATLAB-Based Approach

The MATLAB-based approach involved implementing ANN models using MATLAB commands such as newff, traingd, and newrb. Two training algorithms were used: feedforward gradient descent backpropagation (FFBP) and radial basis function (RBF).

The results indicate that the FFBP algorithm outperformed the RBF algorithm in terms of MSE across all phases of training, validation, and testing.

V. Conclusion

The comparative analysis of artificial neural network (ANN) models for solar radiation forecasting using GUI-based and MATLAB command approaches provides valuable insights into the performance and effectiveness of different methodologies in weather data analysis. This study aimed to evaluate the accuracy, generalization, and computational efficiency of ANN models constructed through these approaches.

The GUI-based approach, employing the nnstart tool and Levenberg-Marquardt backpropagation algorithm, demonstrated the capability to construct ANN models with varying configurations of hidden neurons and training epochs. The results revealed that increasing the number of hidden neurons improved model accuracy, with the model utilizing 100 hidden neurons achieving the lowest Mean Squared Error (MSE) of 18.25, indicating enhanced predictive capabilities.

The MATLAB-based approach utilized MATLAB commands such as newff, traingd, and newrb, employing both feedforward gradient descent backpropagation (FFBP) and radial basis function (RBF) algorithms. The FFBP algorithm outperformed the RBF algorithm in terms of MSE across all phases of training, validation, and testing, showcasing superior learning dynamics and predictive accuracy.

This study contributes to the ongoing research and development efforts in leveraging ANN technologies for improved weather forecasting, paving the way for more accurate and

reliable predictions in the field of meteorology and climate science.

REFERENCES

[1] "Gradient descent backpropagation - matlab traingd," 2022, available: https://www.mathworks.com/help/deeplearning/ref/traingd.html.

[2] "Fit data with a shallow neural network - matlab & simulink," 2022, available: https://www.mathworks.com/help/deeplearning/gs/fit-data-with-a-neural-network.html.

[3] MATLAB and R. LAB, "How to import training and testing data in neural network tool box in matlab?" youTube. Aug. 24, 2022. Available: https://www.youtube.com/watch?v=tXBmdselGzE.

[4] "Climate data online - map search," available: http://www.bom.gov.au/climate/data/.

[5] "North head [wind only], nsw - daily weather observations," available: http://www.bom.gov.au/climate/dwo/IDCJDW2192.latest.shtml.