International Journal of Information, Engineering & Technology
Volume 12, Issue 4, PP 43-67, ISSN: 2360-9194, November, 2023
DOI: 4272-1454-381-246
Double Blind Peer Reviewed International Research Journal
Journal Series: Global Academic Research Consortium (garc)
arcnjournals@gmail.com
https://arcnjournals.org

Application of Feedforward Neural Network Algorithm for a Day Ahead Photovoltaic Power Output Forecast in Maiduguri Metropolis

Bukar Abdullahi [1*], Musa Abdulkadir [2], Mohammed Alkali Abbo[3], Musa Mohammed Gujja[4]
[1]Department of Physics, Federal University Gusau, Nigeria
[2]Department of Electrical and Electronics Engineering, University of Maiduguri, Nigeria
[3]Department of Electrical Engineering Technology, Ramat Polytechnic Maiduguri, Nigeria
[4]Department of Electrical Engineering Technology, Ramat Polytechnic Maiduguri, Nigeria

***Abstract**: This study explores the Application of a Feedforward Neural Network (FFNN) for Day-Ahead Photovoltaic (PV) Power Output Forecasting in Maiduguri Metropolis, a city located in north-eastern Nigeria. Accurate forecasting of PV power output is crucial for efficient integration of solar energy into the electricity grid and optimal energy management. The FFNN model was trained using historical meteorological and PV power generation data collected in Maiduguri. The input variables include temperature, wind speed, humidity, cloud cover, hour of the day, day of the year and previous PV power outputs. The model was designed to predict the PV power output for the day ahead. To evaluate the performance of the FFNN model, various statistical metrics, such as mean absolute percentage error (MAPE) and correlation coefficient (R), are utilized. The model's ability to capture the complex nonlinear relationships between the input variables and the PV power output was assessed. The results demonstrate that the FFNN model is capable of accurately forecasting the day-ahead PV power output in Maiduguri. The obtained MAPE of 8.9093 and R values of 0.7632 by FFNN and MAPE of 15.0048 and R value of 0.3533 by ANFIS which is a validation tool, had indicated that the FFNN suitability in capturing the underlying patterns and dynamics of the PV system. The findings of this research contribute to the advancement of PV power forecasting techniques, using Maiduguri as case study. The FFNN model's demonstrated accuracy and reliability make it a valuable tool for decision-making processes related to renewable energy management, grid stability, and energy market operations.*

***Keywords:** Feed Forward Neural Network (FFNN), APE (Absolute Percentage Error), MAPE (Mean Absolute Percentage error).*

1. INTRODUCTION

Recent years have witnessed a growing emphasis on renewable energy technology as a result of the global search for ecologically friendly and sustainable energy sources. (Rekioua & Matagne, 2012). Among these, solar photovoltaic (PV) systems have emerged as a pivotal solution to harnessing clean energy from the sun. The abundant availability of sunlight in various regions presents a remarkable opportunity to generate electricity without the associated greenhouse gas emissions and depletion of finite fossil fuel resources. The city of Maiduguri, located in the northeastern region of Nigeria, is one such area with substantial solar energy potential. As urbanization and energy demands continue to rise, the integration of photovoltaic power into the energy mix becomes increasingly vital for a sustainable future.

However, the inherent intermittency and variability of solar power production due to changing weather conditions pose challenges to the seamless integration of photovoltaic systems into the electricity grid (Namor, Sossan, Cherkaoui, & Paolone, 2018). These challenges highlight the critical importance of accurate forecasting models that can predict solar power output with precision (Conte, Massucco, Saviozzi, & Silvestro, 2017). Accurate forecasts empower energy system operators, policymakers, and stakeholders to effectively plan and manage the integration of solar energy (Van der Meer, Mouli, Mouli, Elizondo, & Bauer, 2016), ensuring grid stability, efficient energy distribution, and optimal resource allocation.

In this context, the present study endeavors to address the need for reliable solar power output forecasting in Maiduguri by employing advanced artificial intelligence techniques. Specifically, the study focuses on utilizing Feedforward Neural Networks (FFNN), a type of artificial neural network, to develop an accurate forecasting model. Neural networks have demonstrated remarkable capabilities in modeling complex and nonlinear relationships within datasets, making them well-suited for predicting photovoltaic power output based on a multitude of influencing factors.

By looking at data on past solar energy production along with pertinent climatic factors including temperature, wind speed, cloud cover, and relative humidity. This research aims to construct a robust FFNN model that can provide accurate short-term forecasting of PV power output in Maiduguri. The successful development of such a model holds the potential to significantly improve the reliability and efficiency of solar energy integration into the regional electricity grid.

In (Sivaneasan, Yu, & Goh, 2017) proposed an improved solar forecasting algorithm based on artificial neural network (ANN) model with fuzzy logic pre-processing. The proposed model also includes an improved error correction factor aimed at minimizing the forecast error by incorporating the error from previous 5-min forecasted output to the input layer. The clear-sky model and weather data obtained from a weather station in Singapore are used for training the developed model. The numerical result prove that the error correction factor coupled with a pure ANN can significantly improve solar irradiance forecast accuracy due to the adaptive error correction ability. A slight improvement can also be achieved by incorporating a fuzzy logic pre-processing to classify cloud cover index based on relative humidity, rainfall and the time of the day.

In (Nespoli et al., 2019) proposed the analysis the 24-h-ahead power forecasting performance of the two methods; the results show the good forecasting performance of both methods in the case of sunny days. While the hybrid method shows an excellent performance for some specific days, the second method under study shows a more stable and consistently good performance. The forecasting performance of both methods drops significantly for cloudy days. Again, while the performance of the hybrid method is better for some specific days, for at least one day the prediction is rather poor, and the method that feeds exclusively on data from the dataset shows more stable performance on both Normalized Mean Absolute Error (NMAE) and weighted mean absolute error (WMAE) metrics.

In (Aprillia, Yang, & Huang, 2020) proposed a short-term PV power forecasting algorithm based on a CNN-SSA. Convolutional neural network (CNN) regression is used to construct the prediction model, and salp swarm algorithm (SSA) is used to identify the optimal CNN parameter. CNN classification is used for the CNN-SSA to obtain the correct weather type. The results show the proposed method provided better accuracy than the benchmark algorithms did. The proposed algorithm provides a simple approach.

In (Yadav, Pal, & Tripathi, 2019) proposed a new hybrid approach to photovoltaic energy prediction. The proposed approach used Particle swarm optimization (PSO) algorithm to optimize adaptive neuro-fuzzy inference system (ANFIS) parameters. After optimization, the ANFIS parameters were updated and tested. The predicted results of the proposed method were compared with some existing methods BPNN and ANFIS. As a result, it is clear that the predicted result of the proposed method is significant in four seasons. The average % MAPE of the predicted result is 8.42 % of the proposed method. Four-week Symmetric mean absolute percentage error (sMAPE) is much better than previous reference methods, with mean sMAPE being 6.88. Therefore, the proposed method shows the promising result of photovoltaic energy with acceptable computing time.

In (Su, Batzelis, & Pal, 2019), a comprehensive performance assessment among some of the most popular PV power forecasting methods is performed on a common dataset. Non-linear Autoregressive Neural network with exogenous input (NARXNN) is found to be superior over other neural networks due to its dynamic feedback mechanism. Random Forest (RF) performs the best among the intelligence algorithms. There is a seasonal effect on the forecasting problem; summer and autumn are easier to forecast than spring and winter. The training process of a neural network exhibits great randomness, while intelligent algorithms are generally more robust. The proposed Hybrid method performs most favourably among all methods, correcting erroneous fluctuations and negative forecasting. In fact, a major conclusion from this investigation is that simple combination of several good models can generate a more reliable prediction than any single method on its own. This may be found useful especially when there is no complete data for model training.

In (Shi, Zhu, Yuan, Hao, & Wang, 2018), based on the influence of environmental factors on the output power of PV power generation, based on the limit learning machine, a forecasting model of PV power output is set up with weather types, and the historical power data of the PV power station is reasonably divided according to the different weather types, and the weather type is further classified by the grey correlation analysis method. The output results of improved equal-dimension grey (IEDG) model are used as the input of the extreme learning machine (ELM) model, and ELM model is trained and tested. The prediction results are analyzed to get the following conclusions: IEDG-ELM proposed in this paper can effectively predict the output power of PV systems under various weather conditions, IEDG-ELM models proposed in this paper has a relatively accurate prediction ability and strong applicability.

2 Artificial Neural Network (ANN)

Artificial neurons (ANNs) are networks of interconnected nodes that process and send information. These networks are inspired by the arrangement and functioning of biological neural networks found in the human brain. ANNs are capable of learning patterns from data and making predictions or decisions based on those learned patterns.

Each artificial neuron performs a simple computation: it takes a weighted sum of its inputs, applies an activation function, and then passes the result to the next layer of neurons. The connections between neurons have associated weights that are adjusted during the training process to optimize the network's performance on a given task.

The layers of neurons in an ANN are typically organized into three main types:

i. **Input Layer**: This is the first layer and receives the raw input data. Each neuron in this layer corresponds to a feature in the input data.

ii. **Hidden Layers**: These are intermediate layers between the input and output layers. They perform computations on the input data using the learned weights and activation functions. Hidden layers are responsible for extracting relevant features from the data.

iii. **Output Layer**: This is the final layer that produces the network's prediction or output. The number of neurons in this layer depends on the task. For example, in a binary classification task, there would be one neuron to indicate the probability of one class.

2.1 Feed-Forward Neural Network

Feedforward Neural Networks, also known as multi-layer perceptrons (MLPs), are a fundamental type of artificial neural network. They consist of an input layer, one or more hidden layers, and an output layer. Each layer contains a set of interconnected neurons, also known as nodes, and these neurons are organized into a sequential manner where the information flows only in one direction, from the input to the output.

Figure I shows the schematic diagram of the feed-forward neural network (FFNN)

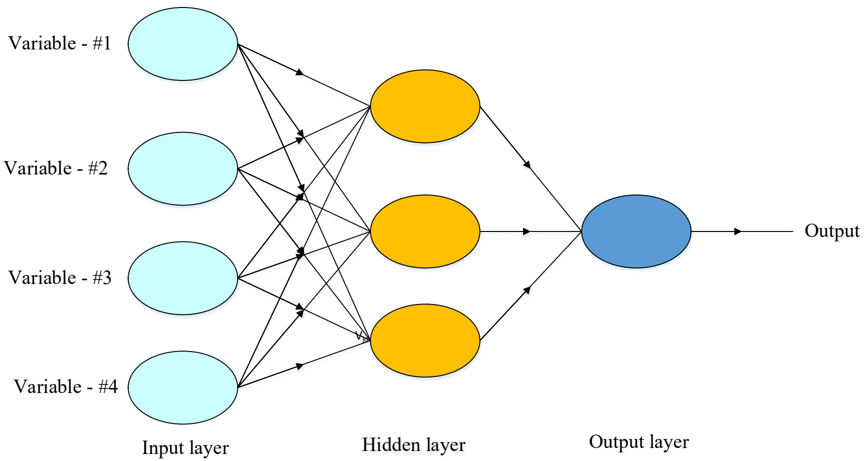

Fig. I: Architecture of FFNN

2.2 Artificial Neural network (ANN) Algorithm

An overview of the mathematics involved in a Feedforward Neural Network (FFNN), including how data flows through the network, the activation functions, and the calculations at each layer.

2.2.1 Data Flow

In a FFNN, data flows through the network in a forward direction, from the input layer to the output layer. Each layer consists of neurons (also called units) that process the input and pass their outputs to the next layer.

2.2.2 Neuron Output

For a neuron in a hidden or output layer, the output can be calculated as follows:

$$Output = Activation(\sum_{i=1}^{n} w_i x_i + b)$$

where:

Activation is the activation function (e.g., ReLU, sigmoid, tanh).
n is the number of inputs.
w_i are the weights associated with each input.
x_i are the input values.
b is the bias term.

2.2.3 Activation Functions

Activation functions introduce non-linearity into the network, enabling it to learn complex relationships in the data. Common activation functions include:

- ReLU (Rectified Linear Activation): $\operatorname{Re}LU(x) = \max(0, x)$

- Sigmoid: $\sigma(x) = \frac{1}{1+e^{-x}}$
- Tanh (Hyperbolic Tangent): $\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$
- Softmax (for output layer in multi-class classification):

  $Soft\max(x)_i = \frac{e^{x_i}}{\sum_{j=1}^{c} e^{x_j}}$ where C is the number of classes.

2.2.4. Forward Propagation

Forward propagation refers to the process of computing the outputs of each layer as data passes through the network. For each layer l, the output O is computed using the formula mentioned earlier:

$$O_j^l = Activation(\sum_{i=1}^{n} w_{ij}^l O_i^{l-1} + b_j^l)$$

2.2.5. Loss Function

The loss function quantifies the difference between the predicted outputs and the true target values.

2.2.6. Backpropagation

Backpropagation involves computing gradients of the loss with respect to the network's parameters (weights and biases). The chain rule is used to propagate gradients backward through the network to update the parameters.

2.2.7. Gradient Descent

Gradient Descent is an optimization algorithm used to update the network's parameters based on the computed gradients. The weights and biases are adjusted in the opposite direction of the gradient to minimize the loss.

2.2.8. Learning Rate

The learning rate controls the step size during gradient descent. It determines how much the parameters are updated in each iteration:

Updated weight = weight – learning rate x Gradient

This provides a basic overview of the mathematics behind a Feedforward Neural Network.

2.3 Performance Indicators of the reliability of the FFNN-based Model

Certainly, when evaluating the performance of a Feedforward Neural Network (FFNN) for prediction tasks, you will want to consider various performance indicators that provide insights into how well the network is making predictions. Here are the indicators used in this research:

2.3.1 Mean Absolute Percentage Error (MAPE)

The MAPE is a metric used to measure the accuracy of predictions in terms of percentage errors. It quantifies the average absolute percentage difference between the predicted values and the actual target values. The formula for MAPE is:

$$MAPE = \frac{1}{N}\sum_{M=1}^{N} abs\left(\frac{PV_{Forecasted}(M) - PV_{Actual}(M)}{PV_{Actual}(M)}\right)\times 100$$

Where,
Abs = absolute value;
N = total number of hours in the testing data (in the case of this work N = 1811)
M stands for the Mth hour in the testing data;
PV$_{forecast}$ (M) = forecasted PV for the Mth hour;
PV$_{actual}$ (M) = actual PV for the Mth hour (this is the PV obtained from solar panel).

2.3.2 Pearson Correlation Coefficient

The Pearson Correlation Coefficient, often denoted as rr, measures the linear relationship between the predicted values and the actual target values. It ranges from -1 to 1, with r = 1 indicating a perfect positive linear correlation. r = −1 indicating a perfect negative linear correlation. r = 0 indicating no linear correlation. The formula for Pearson correlation coefficient is:

$$R^2 = 1 - \sum_i \frac{(a_i - t_i)^2}{(t_i - \overline{t_i})^2}$$

Where, $\overline{t_i}$ mean target value
t$_i$; target outputs
a$_i$; network outputs.

3. METHODOLOGY

3.1 Introduction

This section outlines the implementation of Feedforward Neural Networks (FFNN) as the chosen forecasting technique, detailing the data preprocessing, network architecture, and training process as shown in Figure II. By elucidating the steps taken to develop the FFNN model, this chapter aims to provide a comprehensive understanding of the approach adopted for predicting photovoltaic power output in the specific context of Maiduguri.

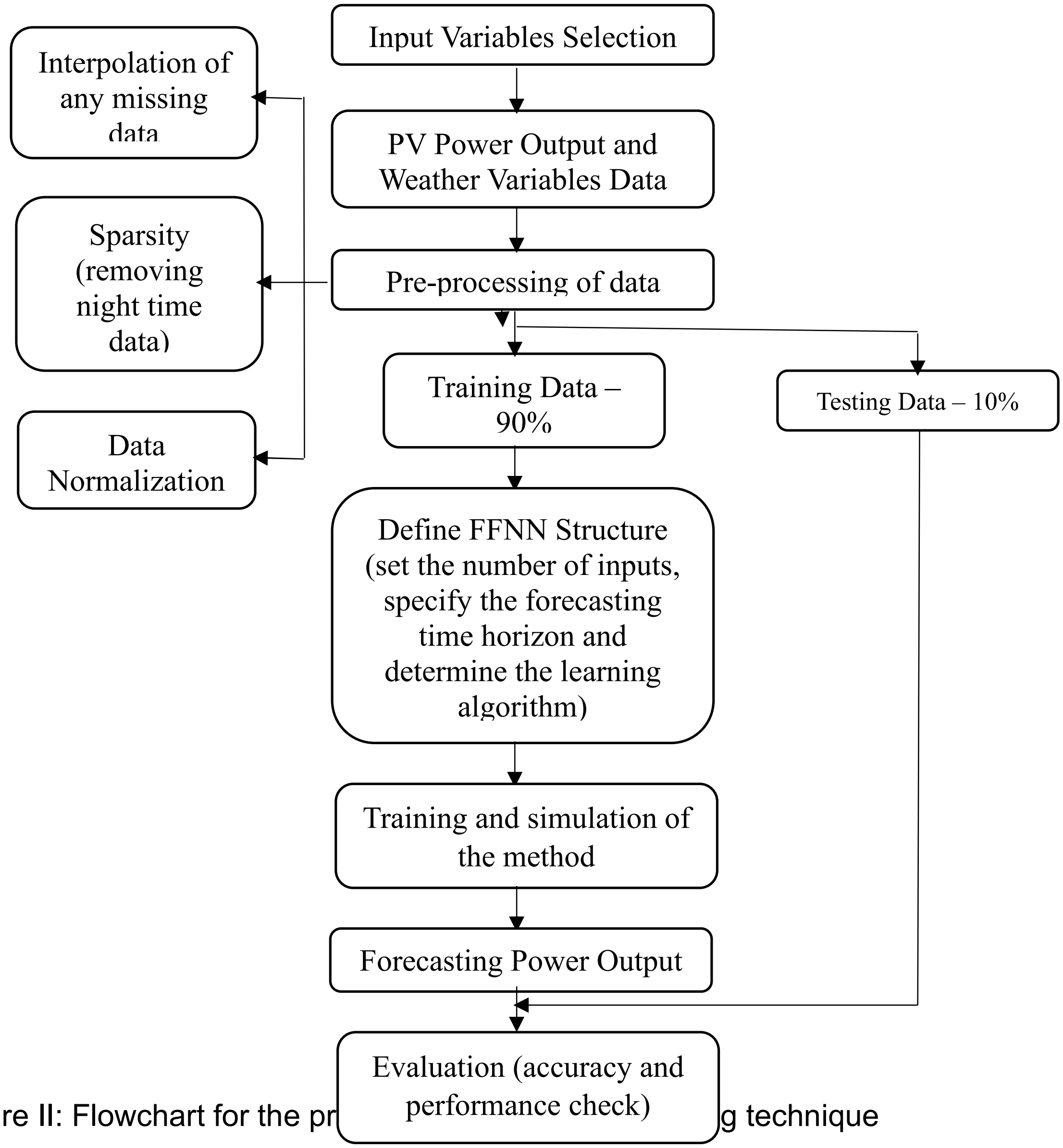
Input Variables Selection
PV Power Output and Weather Variables Data
Pre-processing of data
Interpolation of any missing data
Sparsity (removing night time data)
Data Normalization
Training Data – 90%
Testing Data – 10%
Define FFNN Structure (set the number of inputs, specify the forecasting time horizon and determine the learning algorithm)
Training and simulation of the method
Forecasting Power Output
Evaluation (accuracy and performance check)

Figure II: Flowchart for the pr g technique

3.2 Case Study – The PV Plant

The generated power data was gathered from a solar system that is set up to power water boreholes at Culvert Junction, Indimi Road in Maiduguri Metropolitan Council, which is located 350 meters above sea level and at longitudes 13° 09' East and 11° 51' North. This PV system is shown in Plate 3.1 and has a total power output of 4 KWp and is composed of 12 PV modules of 325W Sunmodule Bisum SW and Inverter. The metrological used were relative humidity, temperature, wind speed and cloud cover which were obtained from Nigerian Metrological Agency (NiMet) website.

Figure III: picture of the plant

The pertinent parameters of the PV panel measured at Standard Test Condition (STC) are as shown in Table I:

Table I: Module Parameters

Parameters	Values
Solar cells	Sunmodule Bisum SW 325 XL duo
Peak power	325W
Rated voltage	37.7V
Rated current	8.68A
Open circuit voltage	47.0V
Short circuit current	9.28A

3.2.1 Selected Inputs and Output of the Model

The FFNN needs a set of data to be trained, validated, and tested. It must have both the network's inputs and the corresponding desired output for a network with supervised learning as the MLP. Since the goal of this work is to anticipate the amount of power produced by a solar system, the generated power itself is the intended output, and the inputs are selected based on the factors that affect its generation. As a result, wind speed, cloud cover, temperature, relative humidity, hour of the day and day of the year were chosen to make up the network's input vector.

3.2.2 Desired Output Vector

DC current and voltage data were necessary for this study. As a result, the PV power (Ppv), which was calculated using the expression in equation 3.1, was used as the required output to the FFNN.

$$P_{pv} = I_{DC} \times V_{DC}$$

A collection of measured data for the months of September 2021 to February 2022 was used to create the desired output vector. Since the PV system only operates during the day and this data only covers the period from 7 o'clock in the morning to 16 o'clock in the afternoon, there would be 10 data points in a day.

Moreover, since the forecast window size selected in this study was one hour, the data was collected in an interval of one hour. Consequently, the number of data points was 10 per day, resulting in 310 data points in one month with 31 days, 300 data points in one month with 30 days and 280 data points in one month with 28 days.

Since September to February was considered, the total number of data points was 1810.

3.2.3 Input Vector

The input vector was deduced from Nigerian Metrological Agency (NiMet) website. These variables were temperature, wind speed, cloud cover and humidity. These variables together with the hour in a day and day in a year composed as the network input vector. Also, it is imperative to notice that the parameters were obtained from the month of September 2021 to February 2022 with 1810 data points for each variable.

The format of days was selected to take into account temporal autocorrelations of the target variable, as suggested in (Ceci, Corizzo, Fumarola, Malerba, & Rashkovska, 2016). Temperature and wind speed were selected because they were involved in penal efficiency estimation. Humidity was included because it influences temperature and irradiance (Mekhilef, Saidur, & Kamalisarvestani, 2012) and it is exploited with interesting results in several literature works (Ogliari, Gandelli, Grimaccia, Leva, & Mussetta, 2016; Zhang et al., 2017), Finally, cloud cover represents a numerical index for the estimation of the sky covering. Notice that Nigerian Meteorological Agency (Nimet) provided all the meteorological inputs. The variables are shown in Table II.

Table II: Arrangement of Input and Output Matrix of 1810 x 7

Hour of the day (h)	**Day of the week**	**Next day temperature (^{0}C)**	**Next day relative humidity (%)**	**Next day wind speed (m/s)**	**Next day cloud cover (%)**	**Actual PV (KW)**
7.00	1.00	25.33	90.00	10.67	43.00	1.50
8.00	1.00	26.67	85.00	11.33	37.00	1.04
..	..	..	..	..	..	..
..	..	..	..	..	..	..
..	..	..	..	..	..	..
..	..	..	..	..	..	..
16.00	181.00	37.67	8.00	22.33	0.00	2.60

3.2.4 Feature Scaling

Variables with different scales are included in the dataset. These situations, where distinct variables may have totally different scales, can result in a false prioritization of some of the variables in the model. As a result, feature scaling of the dataset is done to help speed up the algorithm's calculation and also to increase convergence rates. Less testing time was achieved after the dataset has been trained (Thara, PremaSudha, & Xiong, 2019).

A popular pre-processing technique that was used here is normalization, which lessens the dispersion of the obtained data. In essence, the data are all rescaled so that they all fall within a specific range between 0 and 1. The dataset was normalized by computing:

$$x' = \frac{(x \times 1)}{x_{\max}}$$

Where, x is the observed value and x' is the normalized value.

Literature has demonstrated that normalization has a substantial impact on the output of any model since its primary objective is to assure the quality of the data before it is supplied to any model. (Panigrahi & Behera, 2013). Tables III show the normalized values of the data.

Table III: Normalized values of Input and Output Matrix of 1810 x 7

Hour of the day (h)	**Day of the week**	**Next day temperature (^{0}C)**	**Next day relative humidity (%)**	**Next day wind speed (m/s)**	**Next day cloud cover (%)**	**Actual PV (KW)**
0.4375	0.0055	0.5891	0.9184	0.3333	0.4300	0.3750
0.5000	0.0055	0.6202	0.8673	0.3542	0.3700	0.2600
..	..	..	..	..	..	..
..	..	..	..	..	..	..
..	..	..	..	..	..	..
..	..	..	..	..	..	..
1.0000	1.0000	0.8760	0.0816	0.6979	0.0000	0.6500

3.3 Training, Validation and Test set

Training, validation, and testing are crucial steps in the development of a Feedforward Neural Network (FFNN) for predictive modeling. These procedures aid in making sure the model generalizes properly to fresh, unseen data.

3.3.1. Training

Training involves optimizing the model's parameters (weights and biases) to minimize the difference between its predictions and the actual target values. This is typically done using an optimization algorithm like stochastic gradient descent (SGD). The goal is to find the parameters that minimize a loss function, which measures the discrepancy between predictions and actual targets.

The loss function for a regression problem can be mean squared error (MSE), defined as: $MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$

where:
n is the number of data points.
y_i is the actual target value for the ith data point.
$\hat{y}_i$ is the predicted value for the ith data point.

During training, the weights and biases are updated using the gradient of the loss function with respect to the model parameters. The update rule for SGD is generally given by:

$$parameter_{new} = parameter_{old} - learning\_rate \times gradient$$

where the gradient is computed using the chain rule of calculus.

3.3.2. Validation

Validation is used to tune hyperparameters and assess the model's performance during training. A separate validation dataset, distinct from the training data, is used for this purpose. The model's performance on the validation set helps in preventing overfitting.

Hyperparameters, such as the learning rate, number of hidden layers, and number of neurons in each layer, impact the model's performance. The goal is to select hyperparameters that generalize well to unseen data. Hyperparameter tuning can be performed by evaluating the model's performance on the validation set using different hyperparameter combinations.

3.3.3. Testing

Testing assesses the model's generalization performance on completely unseen data. A separate test dataset, distinct from both the training and validation data, is used. The model's predictions are compared to the actual target values to evaluate its performance.

The most common metrics for regression tasks include:

- Mean Absolute Percentage Error (MAPE):

$$MAPE = \frac{1}{n}\sum \frac{|Actual - Forecast|}{Actual} \times 100$$

- Mean Absolute Error (MAE): $MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$
- Root Mean Squared Error (RMSE): $RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$
- Coefficient of Determination (R^2): $R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$, where $\bar{y}$ is the mean of the actual target values.

3.3.4. Overfitting and Regularization

When a model learns to perform well on training data but cannot generalize to new data, it is said to be overfitting. Regularization techniques, such as L2 regularization (also known as weight decay), can help mitigate overfitting by adding a penalty term to the loss function based on the magnitude of the weights.

L2 regularization adds a term to the loss function:

$$Loss_{regularized} = Loss_{unregularized} + \frac{\lambda}{2}\sum_i \theta_i^2$$

where λ is the regularization parameter and θ_i represents the model parameters.

In summary, training, validation, and testing are essential steps in the development of a FFNN for prediction. These steps ensure that the model learns meaningful patterns from the data, generalizes well to new data, and avoids overfitting. Hyperparameter tuning and regularization play significant roles in achieving a well-performing model.

The topology was selected at random. The maximum number of hidden layers is one, while the maximum number of neurons in the hidden layer is eight. Nevertheless, some parameters were defined, even for the random selection. Those specifications were crucial and supported by the literature. One hidden layer was sufficient to achieve a reasonable generalization for this issue, as illustrated in Figure IV.

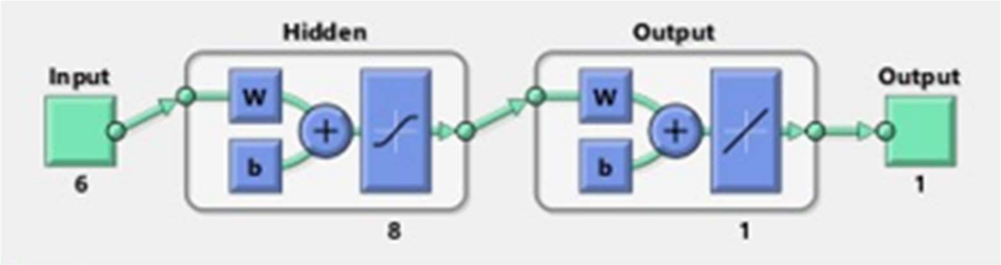

Figure IV: FFNN Model

In this network, the inputs of the network will be time of the day, day of the year, temperature, wind speed, humidity and cloud cover. These parameters were presented to the network and the forecast of the power output of the photovoltaic system was done. The results are shown in the next chapter.

3.4 Training Algorithm

Based on MATLAB's recommendations regarding training time and memory requirements, a backpropagation technique was chosen. one chose to work with and optimize the model based on one training strategy because the model may be trained using other training algorithms. The Levenberg-Marquardt algorithm was selected because it is frequently the quickest backpropagation algorithm in the Deep Learning Toolbox and is strongly suggested as a first-choice supervised approach by (Dias, Antunes, Vieira, & Mota, 2005).

The Levenberg-Marquardt algorithm also known as Levenberg-Marquardt optimization is an iterative optimization method commonly used in nonlinear least squares curve fitting. It is name after Kenneth Levenberg and Donald Marquardt, who independently proposed the algorithm. The algorithm is particularly effective when fitting models with a set of parameters to experimental data by minimizing the sum of the squared differences between the model's predictions and the observed data. The Levenberg-Marquardt algorithms combines the advantages of two other optimization methods: the steepest descent method and the Guass-Newton method. It performs a series of iteration to adjust the parameters of the model until convergence is achieved. During each iteration, the algorithm calculates the Jacobian matrix, which contains the partial derivatives of the model with respect to each parameter. The algorithm then uses the Jacobian matrix, along with the current parameter values and the observed data, to update the parameter estimates. The update step involves solving a system of linear equations, which includes a damping term that combines the steepest descent and Gauss- Newton methods. The damping terms allows the algorithm to converge efficiently even in the presence of ill-conditioned problems. It iterates until a termination requirement, such as attaining a predetermined number of iterations or a desired level of convergence, is satisfied. The result is an estimate of the parameters that minimize the sum of squared differences between the model's predictions and the observed data (Dias et al., 2005).

4. RESULTS AND DISCUSSIONS

4.1 Introduction

This section presents the training of the FFNN, analysis of data collected from PV system and discussion of the result. In this chapter, correlation coefficient R and mean absolute percentage error (MAPE) are used to evaluate the forecasting accuracy of the FFNN Model. The validation of the model was done using ANFIS. Graphical illustrations and analysis of the results was provided using MATLAB GUI.

Figure V shows the total monthly DC power produced by the examined PV system from September 2021 to February 2022. The highest and lowest monthly cumulative powers, 2.5980 KW in February and 1.3557 KW in September, respectively, were obtained.

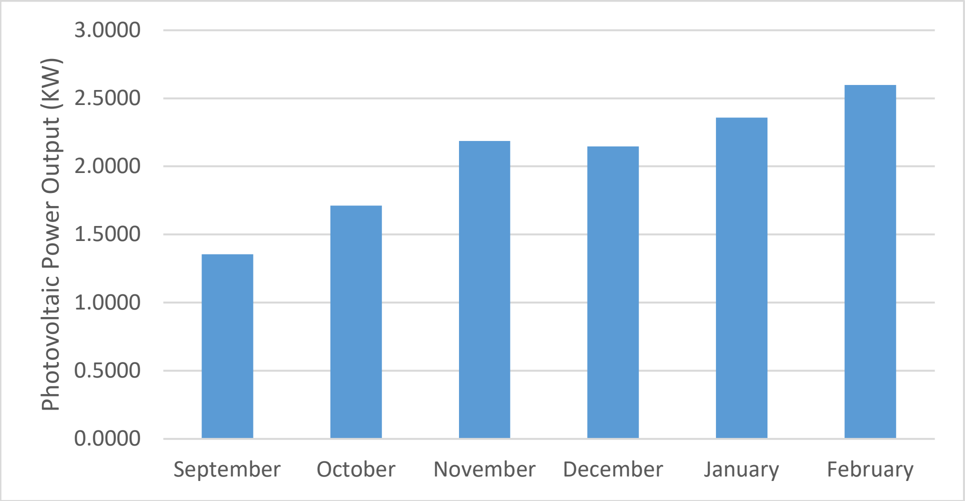

Figure V: Monthly total DC power produced from September 2021 to February 2022.

4.2 Training the FFNN Network

In the context of this work, the Levenberg – Marquardt optimization algorithm was used because of its fast convergence speed. The maximum number of epochs was set to 1000 which is the default value. Learning rate was also set to default value and was allowed to adjust automatically as the training process was in progress.

The results obtained from the simulation of the model were discussed based on three listed plots

- The Regression plot
- The Performance plot
- Training State plot

4.2.1 **Regression Plot**

This comprises of four regression analysis plots as shown in figure VI.

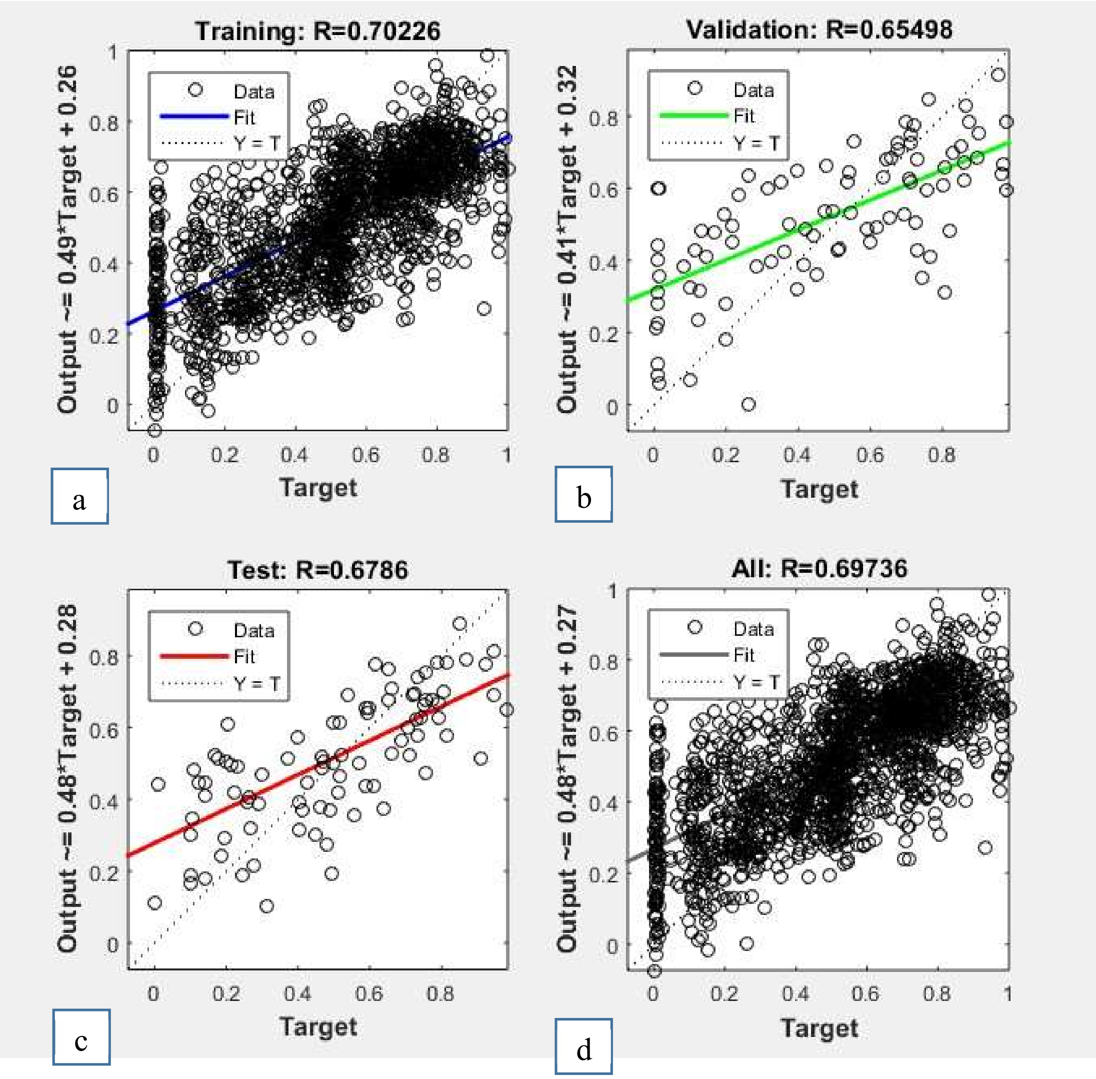

Figure VI: Regression plots.

In a regression plot of a Feedforward Neural Network (FFNN), the R-values represent the correlation coefficients between the predicted outputs and the actual target values for different datasets. Here's what each R-value means:

- **R = 0.70226 for training**: This R-value (0.70226) indicates the correlation between the predicted outputs of the FFNN and the actual target values on the training dataset as shown in Fig. 4.2a. A correlation coefficient of 1.0 would mean a perfect correlation, while a value of 0.0 would indicate no correlation. **R = 0.65498 for validation**: The validation dataset is depicted in Fig. 4.2b, and this R-value (0.65498) shows the correlation between the projected outputs and the actual target values. A stronger correlation is indicated by a higher R-value, which implies that the model's predictions match the validation data more closely.
- **R = 0.6786 for testing**: This R-value (0.6786) denotes the correlation between the predicted outputs and the actual target values on the testing dataset as shown in

Fig. 4.2c. A higher R-value here means that the model's predictions have a relatively strong correlation with the testing data.

- **R = 0.69736 for the overall plot**: This R-value (0.69736) represents the correlation between all predicted outputs and their corresponding actual target values across all datasets combined (training, validation, and testing) as shown in Fig. 4.2d. It provides an overall assessment of the model's performance on the entire dataset.

In summary, the regression plot displays the correlation coefficients (R-values) for the FFNN's predictions compared to the actual target values on different datasets. Higher R-values indicate a better alignment between the predictions and the actual values, suggesting a more accurate and reliable model.

4.2.2 Performance plot

This is a plot of the mean squared error (MSE) against the number of training epochs as shown in Figure VII. From the plot it can be clearly seen that the network had the best validation performance of 0.055679 at epoch 25. The "Best validation performance" of 0.055679 at epoch 25 refers to the lowest value of a performance metric (e.g., Mean Squared Error) achieved by a machine learning model on the validation dataset during its training process up to epoch 25. In this case, the model achieved an MSE of 0.055679 at the 25th training epoch, which indicates that it performed well and had low prediction errors on the validation dataset at that particular point in the training process.

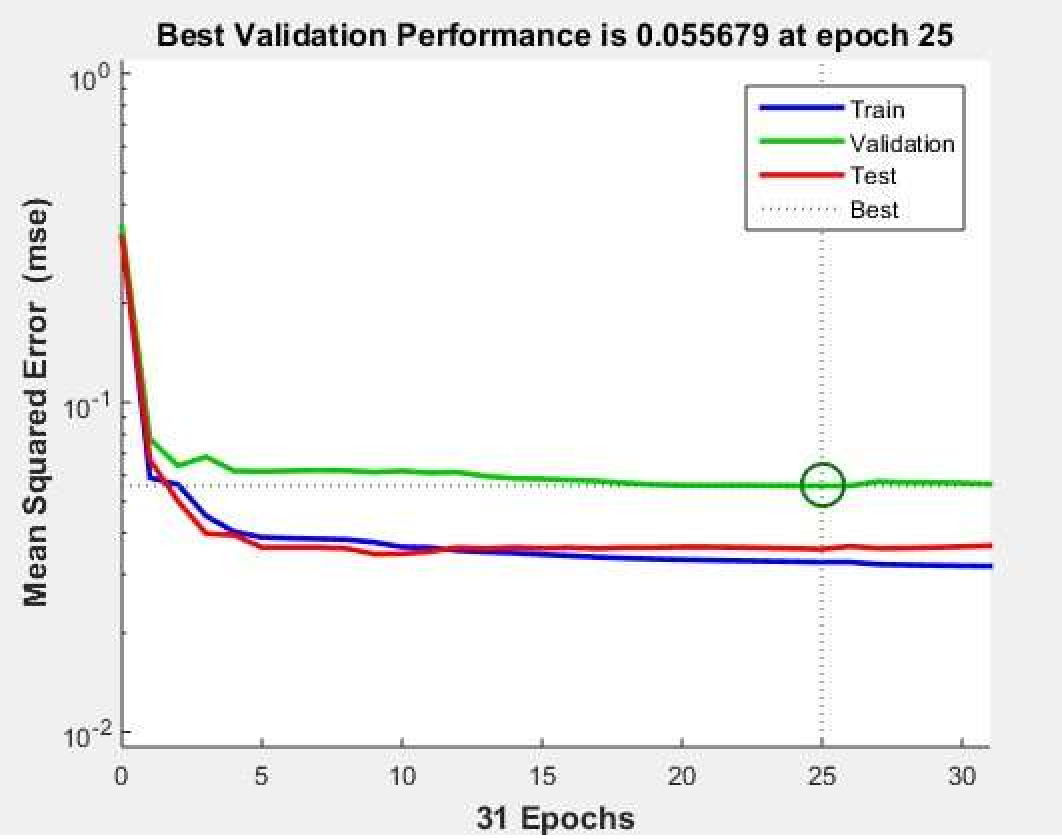

Figure VII: Performance Plot.

4.2.3 Training State Plot

The training state plot comprises of three (3) different plot as shown in Fig. VIII. Each of the plot is explain as follows:

- **Gradient = 0.0052512 at epoch 31**: This indicates the gradient value at epoch 31 during the training process as shown in Fig. 4.4a. The gradient represents the slope of the loss function with respect to the model's parameters. It is used in optimization algorithms, like stochastic gradient descent, to update the model's parameters and minimize the loss.
- **Learning rate = 0.00001 at epoch 31**: As shown in Fig. 4.4b, the learning rate is a hyperparameter that controls the step size during the optimization process. At epoch 31, the learning rate was set to 0.00001, which means the model's parameters were updated using smaller steps, which could lead to slower convergence but may help avoid overshooting the optimal solution.
- **Validation checks = 6 at epoch 31**: As seen in Fig. 4.4c, validation checks are the frequencies at which the model's effectiveness is assessed on the validation dataset during the training phase. At epoch 31, the model was checked on the validation dataset six times, likely after every few epochs, to monitor its performance and prevent overfitting.

In summary, the training state plot provides insights into the training process at epoch 31, including the gradient value, the learning rate used for parameter updates, and the frequency of validation checks to assess the model's performance. These factors collectively influence how the model is optimized and how well it generalizes to unseen data.

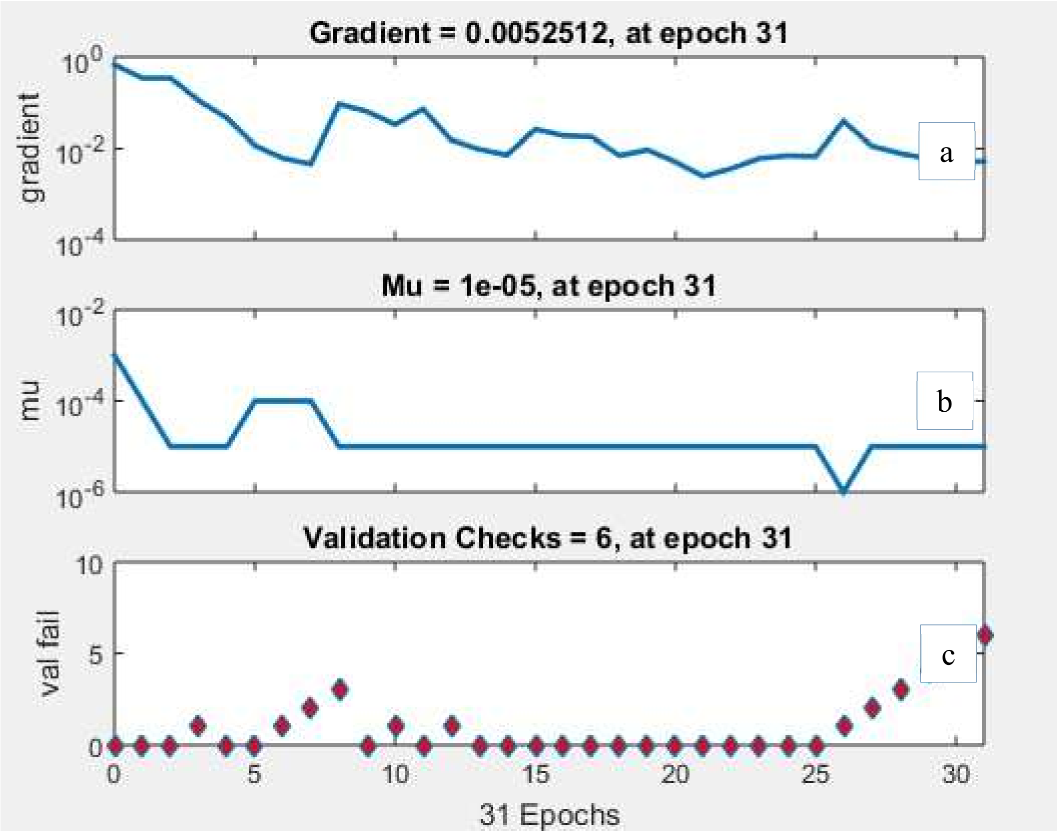

Figure VIII: Training state plot of FFNN model one

4.3 Performance Evaluation

As described in section 2.5, the FFNN was evaluated using MAPE and the result obtained was validated using ANFIS. The result of the performance evaluation was shown in Table IV.

Table IV: Actual and forecast PV of FFNN and ANFIS

FFNN				ANFIS			
Hours	**Actual PV (KW)**	**Forecasted PV (KW)**	**APE**	**Hours**	**Actual PV (KW)**	**Forecasted PV (KW)**	**APE**
7.00	0.7150	0.7097	0.7424	7.00	0.7150	0.5759	19.4545
8.00	0.8625	0.6723	22.0492	8.00	0.8625	0.5831	32.3942
9.00	0.7275	0.6447	11.3793	9.00	0.7275	0.6194	14.8591
10.00	0.6575	0.7048	7.1960	10.00	0.6575	0.7767	18.1293
11.00	0.7550	0.7708	2.0900	11.00	0.7550	0.9000	19.2053
12.00	0.8225	0.8875	7.8986	12.00	0.8225	0.8823	7.2705
13.00	0.8175	0.8516	4.1759	13.00	0.8175	0.7465	8.6850
14.00	0.7675	0.7295	4.9481	14.00	0.7675	0.5972	22.1889
15.00	0.6000	0.6067	1.1233	15.00	0.6000	0.5643	5.9500
16.00	0.5650	0.4097	27.4906	16.00	0.5650	0.5758	1.9115

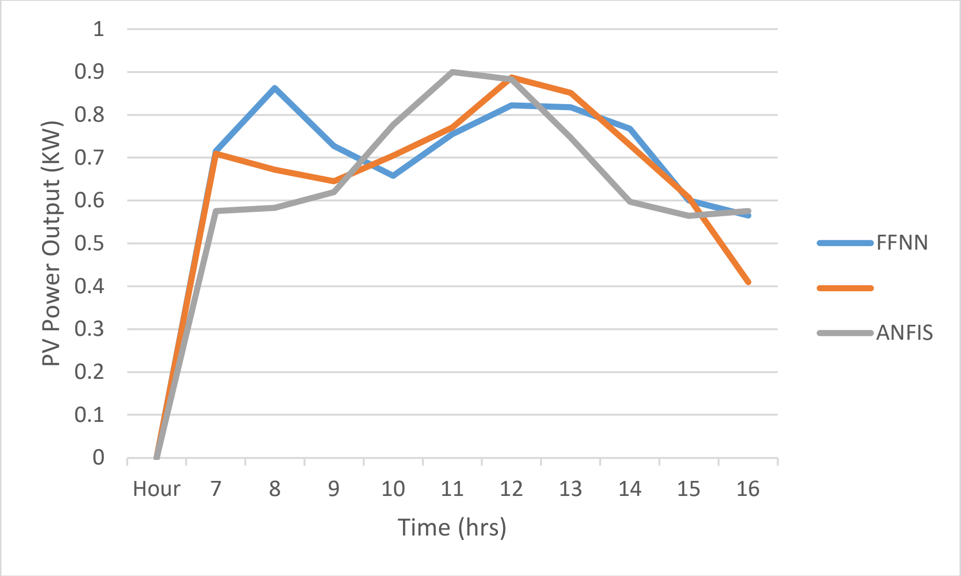

Fig. IX: PV power output Vs hours for a day

As can be inferred from Table 4.1 and also in Fig. IX, the forecasting results of Photovoltaic (PV) power output in Maiduguri using FFNN (Feedforward Neural Network) showed that at 7.00 hours, the predicted power output was 0.7097 kW, with an Absolute Percentage Error (APE) of 0.7424. This indicates that the FFNN model's prediction was quite close to the actual value. On the other hand, the ANFIS (Adaptive Neuro-Fuzzy Inference System) model, used as a validation tool, predicted a power output of 0.7150 kW at 7.00 hours, but its APE was much higher, standing at 19.4545. This implies that the ANFIS model had a larger discrepancy between its predictions and the actual values.

The forecasting results of Photovoltaic (PV) power output using the FFNN at 8.00 hours showed a power output of 0.6723 kW. The APE for this prediction was 22.0492%. On the other hand, the ANFIS as a validation tool predicted a power output of 0.5831 kW at the same time, with an APE of 32.3942%.

The forecasting results of Photovoltaic (PV) power output using the FFNN at 9.00 hours showed a power output of 0.6447 kW. The APE for this prediction was 11.3797%. On the other hand, the ANFIS as a validation tool predicted a power output of 0.6194 kW at the same time, with an APE of 14.8591%. Both models, FFNN and ANFIS, demonstrated relatively low APE values, indicating that they are performing well in predicting the PV power output at 9.00 hours in Maiduguri. The FFNN appears to have a slightly lower APE than ANFIS, suggesting it might have a slightly more accurate prediction for this particular time slot.

The prediction results of Photovoltaic (PV) power output using the FFNN at 10.00 hours showed a power output of 0.7048 kW. The APE for this prediction was 7.1960%. On the other hand, the ANFIS as a validation tool predicted a power output of 0.7767 kW at the same time, with an APE of 18.1293%. In this case, the FFNN demonstrated a relatively low APE, indicating that it provided a more accurate prediction for the PV power output at 10.00 hours in Maiduguri compared to ANFIS. The difference in APE between the two models is significant, with FFNN outperforming ANFIS by a considerable margin in terms of predicting accuracy.

The predicting results of Photovoltaic (PV) power output using the FFNN at 11.00 hours showed a power output of 0.7708 kW. The APE for this prediction was 2.090%. On the other hand, the ANFIS as a validation tool predicted a power output of 0.900 kW at the same time, with an APE of 19.2053%. In this instance, the FFNN demonstrated an extremely low APE, indicating that it provided a highly accurate prediction for the PV power output at 11.00 hours in Maiduguri. On the contrary, ANFIS showed a significantly higher APE, implying that its prediction had a higher percentage of error compared to FFNN. The substantial difference in APE between the two models suggests that FFNN is performing notably better than ANFIS in forecasting the PV power output in Maiduguri at 11.00 hours.

The predicting results of Photovoltaic (PV) power output using the FFNN at 12.00 hours showed a power output of 0.8875 kW. The APE for this prediction was 7.886%. On the other hand, the ANFIS as a validation tool predicted a power output of 0.8823 kW at the same time, with an APE of 7.2706%. In this case, both the FFNN and ANFIS models demonstrated relatively low APE values, indicating that they provided accurate predictions for the PV power output at 12.00 hours in Maiduguri. The difference in APE between the two models is minimal, suggesting that both FFNN and ANFIS performed similarly well in forecasting the power output at this time slot.

The predicting results of Photovoltaic (PV) power output using the FFNN at 13.00 hours showed a power output of 0.8516 kW. The APE for this prediction was 4.1759%. On the other hand, the ANFIS as a validation tool predicted a power output of 0.7465 kW at the same time, with an APE of 8.6859%. In this scenario, the FFNN demonstrated a relatively low APE, indicating that it provided a highly accurate prediction for the PV power output at 13.00 hours in Maiduguri. On the other hand, ANFIS also had a reasonable APE value, although slightly higher than that of FFNN. Overall, both models, FFNN and ANFIS, seem to perform well in forecasting the PV power output at 13.00 hours in Maiduguri. The FFNN shows a slightly better accuracy compared to ANFIS, but the difference in APE between the two models is not significant.

The predicting results of Photovoltaic (PV) power output using the FFNN at 14.00 hours showed a power output of 0.7285 kW. The APE for this prediction was 4.9481%. On the other hand, the ANFIS as a validation tool predicted a power output of 0.5972 kW at the same time, with an APE of 22.1889%. In this case, the FFNN demonstrated a relatively low APE, indicating that it provided a highly accurate prediction for the PV power output at 14.00 hours in Maiduguri. On the contrary, the ANFIS model showed a significantly higher APE, suggesting that its prediction had a higher percentage of error compared to FFNN. The difference in APE between the two models is substantial, with FFNN significantly outperforming ANFIS in forecasting accuracy for the PV power output at 14.00 hours. This suggests that the FFNN model is more reliable and accurate in predicting the power output at this specific time slot in Maiduguri.

The predicting results of Photovoltaic (PV) power output using the FFNN at 15.00 hours showed a power output of 0.6067 kW. The APE for this prediction was 1.1233%. On the other hand, the ANFIS as a validation tool predicted a power output of 0.5643 kW at the same time, with an APE of 5.9500%. In this case, both the FFNN and ANFIS models demonstrated relatively low APE values, indicating that they provided accurate predictions for the PV power output at 15.00 hours in Maiduguri. The FFNN had a slightly lower APE than ANFIS, suggesting that it provided a slightly more accurate prediction for this specific time slot. It's impressive to see both models performing well and providing reliable forecasts for the PV power output. The difference in APE between the two models is minimal, indicating that both FFNN and ANFIS are suitable forecasting tools for PV power output in Maiduguri at 15.00 hours.

The predicting results of Photovoltaic (PV) power output using the FFNN at 16.00 hours showed a power output of 0.4097 kW. The APE for this prediction was 27.4906%. On the other hand, the ANFIS as a validation tool predicted a power output of 0.5758 kW at the same time, with an APE of 1.9115%. In this case, the FFNN demonstrated a high APE, indicating that it provided a less accurate prediction for the PV power output at 16.00 hours in Maiduguri. On the contrary, the ANFIS model showed a significantly lower APE, suggesting that its prediction was more accurate compared to FFNN. The difference in APE between the two models is considerable, with ANFIS outperforming FFNN in forecasting accuracy for the PV power output at 16.00 hours. This suggests that the ANFIS model is more reliable and accurate in predicting the power output at this specific time slot in Maiduguri.

This research presents the results of forecasting solar power output of ahead using the Feedforward Neural Network (FFNN) model. The Mean Absolute Percentage Error (MAPE) of 8.9093 indicates that, on average, the FFNN model's predictions have an error of approximately 8.91%, which suggests a relatively accurate performance in predicting the solar power output.

The R-value of 0.7632 is the correlation coefficient, which measures the strength and direction of the linear relationship between the predicted values and the actual values. An R-value of 0.7632 indicates a moderately strong positive correlation, suggesting that the FFNN model's predictions are reasonably aligned with the actual solar power output.

The research also compares the FFNN model's performance with a validation tool called ANFIS (Adaptive Neuro-Fuzzy Inference System). The ANFIS model has a higher MAPE of 15.0048, indicating that, on average, its predictions have a larger error of approximately 15.00% compared to the FFNN model.

Moreover, the R-value of 0.3533 for ANFIS shows a weak positive correlation between the predicted values and the actual values, suggesting that ANFIS's predictions are less accurate and less aligned with the actual solar power output compared to the FFNN model.

Based on the results, the research suggests that the FFNN model performs better in forecasting the Photovoltaic power output in Maiduguri compared to the ANFIS model. It demonstrates a lower MAPE and a stronger correlation with the actual values, indicating its superiority in predicting solar power generation in this specific scenario.

5. Conclusion

A Feedforward Neural Network (FFNN) model was trained using historical PV power output, meteorological data (such as temperature, relative humidity, wind speed, and cloud cover), and other pertinent factors (such as hour of the day and day of the year). This comprehensive dataset was employed to develop a robust FFNN model for potential applications in forecasting PV power output based on meteorological conditions and time-related factors.

The collected data, including historical PV power output, meteorological variables (temperature, relative humidity, wind speed, cloud cover), and time-related factors (hour of the day, day of the year), underwent a preprocessing phase. This preprocessing aimed to transform and condition the data, making it suitable for use as input to the Feedforward Neural Network (FFNN) model. By preparing the data appropriately, it ensured that the FFNN model could effectively learn and make accurate predictions regarding PV power output based on the provided meteorological and temporal information.

A Feedforward Neural Network (FFNN) architecture was successfully implemented. This FFNN architecture was designed specifically for time series forecasting, with the primary goal of predicting PV power output. By tailoring the FFNN to handle time-related patterns and relationships, it laid the foundation for accurate and effective predictions of PV power output based on the given historical data, meteorological variables, and relevant temporal factors.

The implemented Feedforward Neural Network (FFNN) model was trained using historical data. The training process involved feeding the FFNN with the collected dataset, which included historical PV power output, meteorological variables (temperature, relative humidity, wind speed, cloud cover), and time-related factors (hour of the day and day of the year). Through this training, the FFNN learned the underlying patterns and relationships within the data, enabling it to make accurate predictions of PV power output based on future meteorological conditions and temporal factors.

The trained Feedforward Neural Network (FFNN) model was successfully implemented for the purpose of day ahead PV power output forecasting in the Maiduguri metropolis. The FFNN model, which had been trained using historical data comprising PV power output, meteorological variables, and relevant temporal factors, was now utilized to predict PV power output for a day ahead in Muhammadu Road of Maiduguri. This implementation has provided valuable insights into the expected PV power generation based on the anticipated meteorological conditions, enabling better resource planning and decision-making for energy management in the Maiduguri metropolis.

In conclusion, the research investigated the forecasting of Photovoltaic (PV) power output in Maiduguri using a Feedforward Neural Network (FFNN) model. The FFNN model demonstrated superior performance with a lower Mean Absolute Percentage Error (MAPE) of 8.9093 and a higher correlation coefficient (R) of 0.7632 compared to the ANFIS (Adaptive Neuro-Fuzzy Inference System) validation tool, which yielded a MAPE of 15.0048 and an R value of 0.3534. This confirms the efficacy of the FFNN model in accurately predicting PV power output and highlights its potential for enhancing renewable energy forecasting in Maiduguri.

6. ACKNOWLEDGEMENT

Our sincere gratitude goes to Engr. Dr. Bukar Baba Aji FNSE for the financial and logistic support given to us during this research.

REFERENCES

Aprillia, H., Yang, H.-T., & Huang, C.-M. (2020). Short-term photovoltaic power forecasting using a convolutional neural network–salp swarm algorithm. *Energies, 13*(8), 1879.

Ceci, M., Corizzo, R., Fumarola, F., Malerba, D., & Rashkovska, A. (2016). Predictive modeling of PV energy production: How to set up the learning task for a better prediction? *IEEE transactions on industrial informatics, 13*(3), 956-966.

Conte, F., Massucco, S., Saviozzi, M., & Silvestro, F. (2017). A stochastic optimization method for planning and real-time control of integrated pv-storage systems: Design and experimental validation. *IEEE Transactions on Sustainable Energy, 9*(3), 1188-1197.

Dias, F. M., Antunes, A., Vieira, J., & Mota, A. M. (2005). *On-line training of neural networks: a sliding window approach for the Levenberg-Marquardt algorithm.* Paper presented at the Artificial Intelligence and Knowledge Engineering Applications: A Bioinspired Approach: First International Work-Conference on the Interplay Between Natural and Artificial Computation, IWINAC 2005, Las Palmas, Canary Islands, Spain, June 15-18, 2005, Proceedings, Part II 1.

Mekhilef, S., Saidur, R., & Kamalisarvestani, M. (2012). Effect of dust, humidity and air velocity on efficiency of photovoltaic cells. *Renewable and Sustainable Energy Reviews, 16*(5), 2920-2925.

Namor, E., Sossan, F., Cherkaoui, R., & Paolone, M. (2018). Control of battery storage systems for the simultaneous provision of multiple services. *IEEE Transactions on Smart Grid, 10*(3), 2799-2808.

Nespoli, A., Ogliari, E., Leva, S., Massi Pavan, A., Mellit, A., Lughi, V., & Dolara, A. (2019). Day-ahead photovoltaic forecasting: A comparison of the most effective techniques. *Energies, 12*(9), 1621.

Ogliari, E., Gandelli, A., Grimaccia, F., Leva, S., & Mussetta, M. (2016). *Neural forecasting of the day-ahead hourly power curve of a photovoltaic plant.* Paper presented at the 2016 International Joint Conference on Neural Networks (IJCNN).

Panigrahi, S., & Behera, H. (2013). Effect of normalization techniques on univariate time series forecasting using evolutionary higher order neural network. *International Journal of Engineering and Advanced Technology, 3*(2), 280-285.

Rekioua, D., & Matagne, E. (2012). *Optimization of photovoltaic power systems: modelization, simulation and control*: Springer Science & Business Media.

Shi, N., Zhu, X., Yuan, P., Hao, J., & Wang, Q. (2018). *Photovoltaic output power prediction based on weather type.* Paper presented at the IOP Conference Series: Earth and Environmental Science.

Sivaneasan, B., Yu, C., & Goh, K. (2017). Solar forecasting using ANN with fuzzy logic pre-processing. *Energy procedia, 143*, 727-732.

Su, D., Batzelis, E., & Pal, B. (2019). *Machine learning algorithms in forecasting of photovoltaic power generation.* Paper presented at the 2019 International Conference on Smart Energy Systems and Technologies (SEST).

Thara, D., PremaSudha, B., & Xiong, F. (2019). Auto-detection of epileptic seizure events using deep neural network with different feature scaling techniques. *Pattern Recognition Letters, 128*, 544-550.

Van der Meer, D., Mouli, G. R. C., Mouli, G. M.-E., Elizondo, L. R., & Bauer, P. (2016). Energy management system with PV power forecast to optimally charge EVs at the workplace. *IEEE transactions on industrial informatics, 14*(1), 311-320.

Yadav, H. K., Pal, Y., & Tripathi, M. M. (2019). PSO tuned ANFIS model for short term photovoltaic power forecasting. *Int. J. Recent Technol. Eng, 7*(6), 937-942.

Zhang, X., Jiang, B., Zhang, X., Fang, F., Gao, Z., & Feng, T. (2017). *Solar photovoltaic power prediction based on similar day approach.* Paper presented at the 2017 36th Chinese Control Conference (CCC).