2023 IEEE International Conference on Power Electronics, Smart Grid, and Renewable Energy (PESGRE) | 979-8-3503-1057-3/23/$31.00 ©2023 IEEE | DOI: 10.1109/PESGRE58662.2023.10404314

Short-Term Photovoltaic Power Forecasting Using Convolution Neural Network

Teja Ginjala
Dept. of Electrical Engineering
NIT Warangal
Warangal, India
tgee21210@student.nitw.ac.in

A. V. Giridhar
Dept. of Electrical Engineering
NIT Warangal
Warangal, India
giridhar@nitw.ac.in

Kirshna Reddy Konda
ZF TCI
Hyderabad, India
krishna.konda@zf.com

Srinivas Yalagam
ZF TCI
Hyderabad, India
srinivas.yalagam@zf.com

***Abstract*—This work presents the four types of short-term photovoltaic (PV) power forecasting models using a 1D convolution neural network based on 4 types of data handling. Multi-Timestamp Multi-Feature-2 (MTMF-2) forecasting model gives satisfactory results compared to Single-Timestamp Multi-Feature (STMF) forecasting, Multi-Timestamp Multi-Feature-1 (MTMF-1) forecasting, and Multi-Timestamp Single-Feature (MTSF) forecasting. Historical data taken from one of the PV stations in China is used as input to these forecasting models. Data contains all 6 features which are Power, Irradiance, Temperature, Pressure, Wind Direction, and Wind Speed for every 15-minute timestamp. TensorFlow libraries are used for model building, and the results are taken from the VS code IDE.**

***Index Terms*—Convolution neural network, Data handling, Multi features, Photovoltaic, Short-Term, Timestamp**

I. Introduction

The penetration of renewable energies has increased during these last decades since it has become an effective solution to the world's energy challenges. Solar energy is the most prominent energy source among all renewable energy sources [1]. By using PV technology, we are converting solar energy into electrical energy. The generated electrical energy depends on different factors like irradiance, temperature, wind direction, wind speed, and pressure [2]. All these factors are intermittent, so the power generated by PV stations is also intermittent. The intermittent power penetration to the grid gives economic and operational challenges to the power system. The accurate forecasting of Renewable energy generation can help in Unit commitment, regulate power, and power quality [3].

Today, the study of PV forecasting is becoming popular in terms of prediction research territory. There are different types of forecasting based on time period [2].

- Short-Term(1hour-24hours):Used for unit commitment,control spring reserves.
- Medium-Term(1day-1week,1week-1month):Crucial for efficient operation and maintenance.
- Long-Term(1month-1year):Important for planning and design of generation transmission and distribution systems for future demand.

Many researchers focus on single-timestamp multi-feature prediction [4], which involves predicting all features of next timestamp. However, this approach typically does not involve predicting exactly one day ahead. This work mainly concentrates on multi-timestamp prediction by handling the data for training and testing. This multi-timestamp prediction gives more error compared to the single-timestamp prediction, but practically multiple time stamp forecasting is the correct way of forecasting the features and power. Data analysis, convolution neural network model, training the model and results are discussed in this paper.

II. Data analysis

The power output of a photovoltaic (PV) system is influenced by weather conditions, so it is necessary to gather historical power data and weather data at the same time. The data is collected from the inverter and weather devices simultaneously. In this work we used PVOD dataset [5], which consists of data from 10 PV stations and includes features such as irradiance, temperature, pressure, wind direction, wind speed, and power, all recorded at 15-minute intervals. Data from one of the PV stations was used for this work. Input data was in the form of a time sequence. All 6 features of time sequence data are shown in Fig. 1.

The data in this work consists of time series information, which means that it is collected and recorded over time. To prepare the data for model training and testing, the window technique method [4] was used.In the window technique, a sliding window of a fixed size is applied to the time series data. The window moves step by step through the data, and at each step, a subset of the data within the window is considered as one input sample for the model.

All features were normalized and considered as one vector 'X' [X=P(Power), T(Temperature), Pr (Pressure), WD (Wind Direction), WS (Wind Speed)] at each timestamp (TS) is shown in TABLE I at a particular date and time (DT). In this work, four types of data patterns were created which are shown below

TABLE I
SAMPLE DATA

TS	DT	I W/m2	T ○C	Pr hPa	WD degrees	WS m/s	P MW
0	8/15/2018 16:00	0	25.90	1006.299988	353	1.1	0
1	8/15/2018 16:15	0	25.90	1006.200012	330	0.9	0

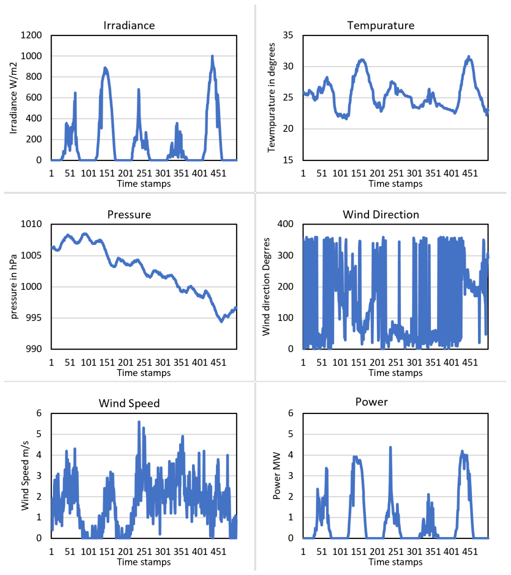

Figure 1. Data visualization

TS- Time Stamp, DT- Date Time, I- Irradiance, T- Temperature, Pr- Pressure, WD- Wind direction, WS- Wind speed, P- Power

A. Single-Timestamp Multi-Feature (STMF) forecasting data preparation

STMF data handling is used for all features of single timestamp data forecasting. The input data is taken as fixed window-sized timestamps of all feature data. The predicted output (all features) is only used for evaluation purposes, not for subsequent predictions. This time sequence data handling is shown in TABLE II for a window size of 5. After data handling, some patterns are used for training and the remaining patterns are used for testing. This data handling method is only used for single time stamp multi-feature forecasting because both training data pattern preparation and testing data pattern preparation are same.

TABLE II
TYPE 1 TRAINING

Input	Output
X_0, X_1, X_2, X_3, X_4	X_5
X_1, X_2, X_3, X_4, X_5	X_6
X_2, X_3, X_4, X_5, X_6	X_7

B. Multi-Timestamp Multi-Feature-1 (MTMF-1) forecasting data preparation

This type of data handling is also used for time series forecasting. Training data patterns are the same as STMF but during testing, the predicted output is used as input for the next prediction, forming a sequence of predictions. The preparation of the testing data set is illustrated in TABLE III assuming testing data starts from the Nth timestamp and here window size is 5. Predicted output values are represented by **X** (bold X).

TABLE III
TYPE 2 TESTING

Input	Output
$X_n, X_{n+1}, X_{n+2}, X_{n+3}, X_{n+4}$	$\mathbf{X_{n+5}}$
$X_{n+1}, X_{n+2}, X_{n+3}, X_{n+4}, \mathbf{X_{n+5}}$	$\mathbf{X_{n+6}}$
$X_{n+2}, X_{n+3}, X_{n+4}, \mathbf{X_{n+5}}, \mathbf{X_{n+6}}$	$\mathbf{X_{n+7}}$

C. Multi-Timestamp Single-Feature (MTSF) forecasting data preparation

MTSF data handling is also used for one feature time series forecasting. The input data is window-sized timestamps of all feature data, and the output is n time stamps of power (P). This model can directly predict the required number of hours of power. For instance, when the window size is 700, it represents nearly one week of data as input. And if n is 100, it means one day of power is given as output. This is demonstrated in TABLE IV, where both the training and testing data sets are prepared in the same way.

TABLE IV
TYPE 3 DATA SAMPLING

Input	Output
$X_0, X_1, X_2, X_3.....X_{698}, X_{699}$	$P_{700}, P_{701}, ..., P_{798}, P_{799}$
$X_1, X_2, X_3, X_4.....X_{699}, X_{700}$	$P_{701}, P_{702}, ..., P_{799}, P_{800}$
$X_2, X_3, X_4, X_5, ...X_{700}, X_{701}$	$P_{702}, P_{703}, ..., P_{800}, P_{801}$

D. Multi-Timestamp Multi-Feature-2 (MTMF-2) forecasting data preparation

This type of data preparation is also used for multi-feature time series forecasting. In this type, a window size of past data pointing at a particular time is used as input, and the output is the next data point at the same time instant. For example, if the window size is 2, the data preparation is shown in TABLE V. This model also gives the multi-timestamp of all feature output. Both training data pattern preparation and testing data pattern preparation are same.

TABLE V
TYPE 4 DATA SAMPLING

Time	Input	Output
16:00	$X_0(8/15/2018), X_{96}(8/16/2018)$	$X_{192}(8/17/2018)$
16:15	$X_1(8/15/2018), X_{97}(8/16/2018)$	$X_{193}(8/17/2018)$
16:30	$X_2(8/15/2018), X_{98}(8/16/2018)$	$X_{194}(8/17/2018)$

III. CONVOLUTION NUERAL NETWORK MODEL(CNN)

Convolutional Neural Network (CNN) is a powerful tool for processing multi-dimensional data [6], particularly in image recognition tasks. They consist of multiple layers including convolutional, pooling, flattening, and dense layers, that extract, and process features from the input data [7]. The convolutional layer applies filters to the input data to extract features, while pooling layers reduce the spatial dimensions of the data for efficient feature extraction. The dense layers at the end of the network make predictions based on the processed features. The network is trained using the backpropagation algorithm, updating the filter weights to minimize the error between the predicted and target outputs.

A. Single-Timestamp Multi-Feature (STMF) forecasting Model

In this study, STMF data patterns were used, which are shown in Table II for training and testing a model. The aim of this model is to predict the next time instant of all features (power, irradiance, pressure, temperature, wind direction, wind speed). The model architecture is shown in Fig. 2. The model has low error as it only predicts the next time instant. However, it doesn't provide forecasting one day ahead.

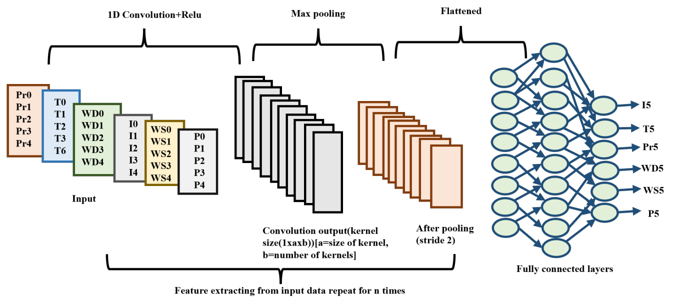

Figure 2. Single-Timestamp Multi-Feature (STMF) forecasting Model

B. Multi-Timestamp Multi-Feature-1 (MTMF-1) forecasting Model

The architecture of the MTMF-1 model is same as that of the STMF model. This model is trained as same as the STMF model, but testing is done by the MTMF-1 testing data pattern shown in TABLE III. However, the use of predicted values for making the next prediction results in a higher error compared to the STMF model. This is due to the accumulation of errors from the previous predictions that deviate the model from the original curve, as demonstrated by the results.

C. Multi-Timestamp Single-Feature (MTSF) forecasting model

The limitations of the STMF and MTMF-1 models are addressed by using MTSF. The model uses MTSF data patterns shown in TABLE IV for training and testing. For one-day ahead prediction, 100-time stamps are used as outputs because each timestamp is 15 minutes. This model results in an error that is higher than that of the STMF model but lower than that of the MTMF-1 model. It provides a sense of the one-day ahead prediction. The model architecture is shown in Fig. 3.

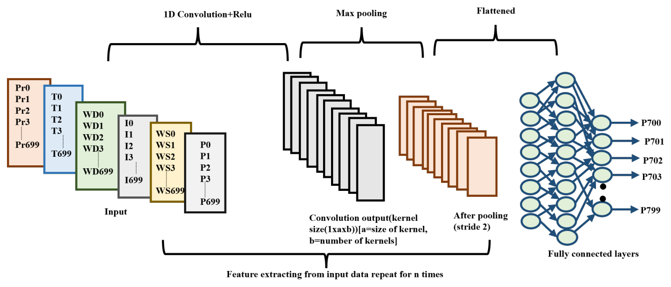

Figure 3. Multi-Timestamp Single-Feature (MTSF) forecasting model

D. Multi-Timestamp Multi-Feature forecasting-2(MTMF-2) Model

This model utilizes the STMF-2 data set and takes past data into consideration from the same time to make predictions for the next day's output at the same time. The results of our analysis showed by this model outperformed compared to above MTSF. The architecture is shown in Fig. 4. This model predicts all six features. This model gives a high error compared to the STMF model, but it gives a practical sense of forecasting.

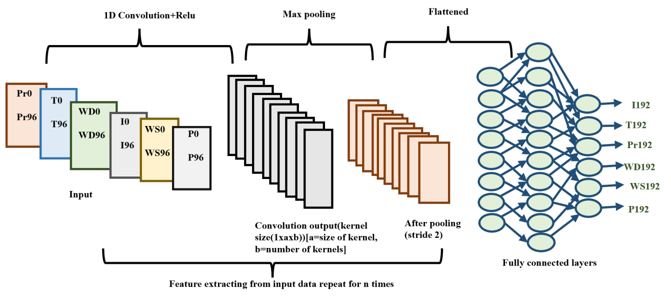

Figure 4. Multi-Timestamp Multi-Feature forecasting-2(STMF-2) Model

IV. TRAINING THE MODEL

Training the models involves adjusting the hyperparameters to optimize the learning parameters, leading to reduced error. There are various types of hyperparameters that can be tuned.

A. Number of convolution layers

In a Convolutional Neural Network (CNN), the convolution layer is one of the key hyperparameters that can significantly impact the model's performance. This layer performs a mathematical operation, called convolution, to extract important features from the input data. The hyperparameters that can be adjusted in a convolution layer include the number of filters, the size of filters, and the stride size. The number of filters determines the depth of the layer and the number of features that can be extracted. The size of filters controls the scope of features captured by each filter, while the stride size determines how the filters slide over the input data. By adjusting these hyperparameters, the model can be fine-tuned to improve its accuracy and efficiency.

B. Activation function

Activation functions play an important role in deep learning as they introduce non-linearity in the model. The activation function is used to determine the output of each neuron in a neural network. The choice of activation function is a crucial hyperparameter as it affects the network's ability to learn and make predictions. Common activation functions used in deep learning include sigmoid, ReLU, tanh, and softmax. The choice of activation function should be based on the specific problem being solved and the properties of the data being used. The ReLU activation function, for instance, is widely used for its computational efficiency and the ability to handle sparse data. The sigmoid activation function is useful for binary classification problems, while the softmax activation function is suitable for multiclass classification problems. Thus, the selection of activation functions should be carefully considered while tuning the hyperparameters in a deep learning model.

C. Learning rate

The learning rate is a scalar value that determines the step size of the updates made to the model parameters during training.The learning rate is a crucial hyperparameter that influences the convergence and performance of a machine learning model. A small learning rate results in slow convergence while a large learning rate may cause overshooting of the optimal solution.The appropriate learning rate is dependent on the specific problem and the model architecture and must be selected through experimentation and validation.

D. Loss function

The loss function is a crucial hyperparameter in machine learning that measures the difference between the predicted output and the true output. It is used to guide the optimization process during training and determine the model's performance. The appropriate loss function depends on the specific problem and the model architecture. Common loss functions include mean squared error, categorical cross-entropy, and binary cross-entropy. The squared error loss function was used in this work.

E. Optimizer

An optimizer is an algorithm used to adjust the model parameters during training in order to minimize the loss. The choice of optimizer can significantly impact the convergence and performance of a machine learning model. There are various types of optimizers available, including gradient descent, stochastic gradient descent, Adam [8], and many others. Each optimizer has its own strengths and weaknesses,and the appropriate optimizer depends on the specific problem and model architecture. Some optimizers have hyperparameters that can be adjusted to further fine-tune performance, such as the learning rate and momentum. Optimizers are implemented as part of the training process and are typically provided as part of deep learning frameworks, such as TensorFlow. In this model, we used the Adam optimizer for better performance.

V. RESULTS DISCUSSION

In this section, the results of each model were discussed and compared. The performance of each model was evaluated and analyzed, and the strengths and limitations of each model were highlighted. The tables in this section represent the testing results of each model for different hyperparameters. This information was used to determine which model was the best for given task of predicting power. STMF model predictions have low error as only one instance is predicted. Fig. 5 represents the results of these predictions. The next instance of all 6 features Power, temperature, Pressure, irradiance, wind direction, and wind speed are predicted. TABLE VI represents the results of different hyperparameters for the STMF model. The results show the impact of changing the values of hyperparameters on the performance of the model. The table provides a comparison of the performance of the model with different hyperparameters and helps to determine the optimal set of hyperparameters for the problem being considered.

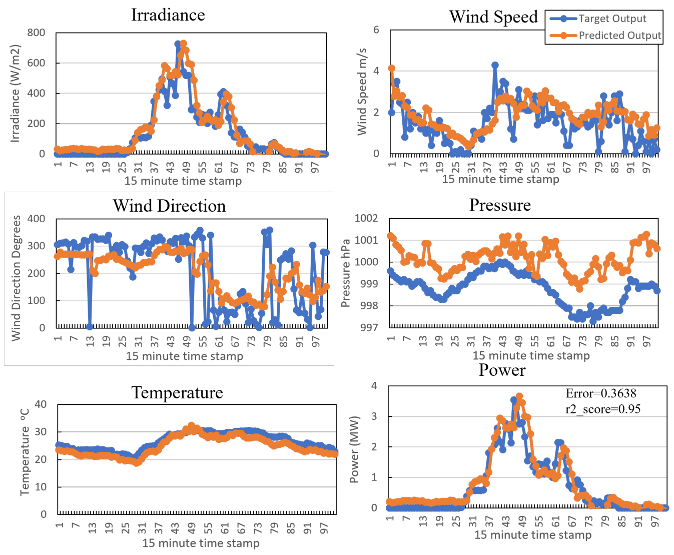

Figure 5. STMF Prediction

Fig. 6 represents the test results of the MTMF-1 model. In this model, the previous predicted value is used for the next prediction. If the first prediction has some error, the same prediction is used for the next prediction, resulting in an increment in error. This figure illustrates the impact of error accumulation on the performance of the model when the previous predicted value is used in subsequent predictions.

TABLE VI
STMF MODEL TESTING RESULTS FOR DIFFERENT HYPERPARAMETERS RELU AS ACTIVATION

Window	Convolution	Filters	Learning rate	activation	Epoch	RMSE Test
10	1	10	0.001	ReLU	200	0.2979
10	1	10	0.005	ReLU	80	0.3686
20	1	10	0.001	ReLU	200	0.3053
20	2	10,10	0.001	ReLU	80	0.3344
50	2	50,30	0.001	ReLU	80	0.3230
70	3	50,30,10	0.001	ReLU	80	0.3780
10	1	10	0.001	Tanh	80	0.3853
50	1	10	0.005	Tanh	80	0.3853
50	1	10	0.005	sigmoid	80	0.3588

TABLE VII
MTSF MODEL TESTING RESULTS FOR DIFFERENT HYPERPARAMETERS RELU AS ACTIVATION

Window	Convolution	Filters	Learning rate	activation	Epoch	RMSE Test
700	1	20	0.001	ReLU	150	0.9837
700	2	20,20	0.001	ReLU	150	1.0417
700	4	20,20,20,20	0.001	ReLU	80	1.6323
700	4	20,20,20,20	0.001	ReLU	80	0.6323
700	6	10,10,10,7,10,3	0.001	ReLU	150	0.8859
800	4	20,20,20,20	0.001	ReLU	805	01.040
500	1	20	0.001	ReLU	80	1.166
700	8	20,20,2,10,20,20,7,10,10	0.001	ReLU	100	0.633
700	11	10,10,10,10,10,7,10,10,7,10,7	0.001	ReLU	100	0.8514
700	8	5,10,10,5,10,7,10,17	0.001	ReLU	100	0.8216

TABLE VIII
MTMF 2 M ODEL T ESTING R ESULTS OR DIFFERENT HYPERPARAMETERS RELU AS ACTIVATION

Window	Convolution	Filters	Learning rate	activation	Epoch	RMSE Test
5	2	10,7	0.001	ReLU	100	0.551
5	4	4	0.001	ReLU	100	0.5913
8	4	10,7,5,3	0.001	ReLU	100	0.5358
14	4	10,7,5,3	0.001	ReLU	100	0.6019
8	2	10,7	0.0001	ReLU	150	0.6811
21	3	15,7,5	0.001	ReLU	200	0.4269
28	2	15,15	0.001	ReLU	200	0.4902
30	3	15,7,15	0.001	ReLU	200	0.5875

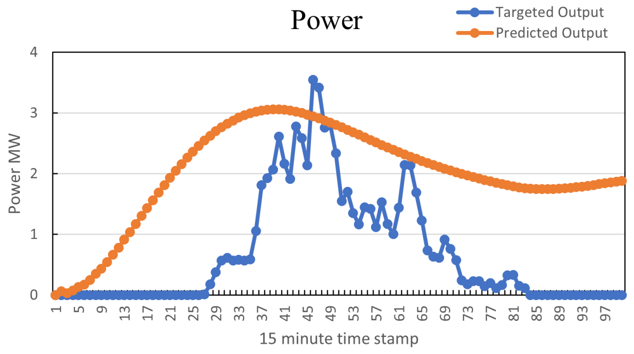

Figure 6. MTMF-1 Prediction

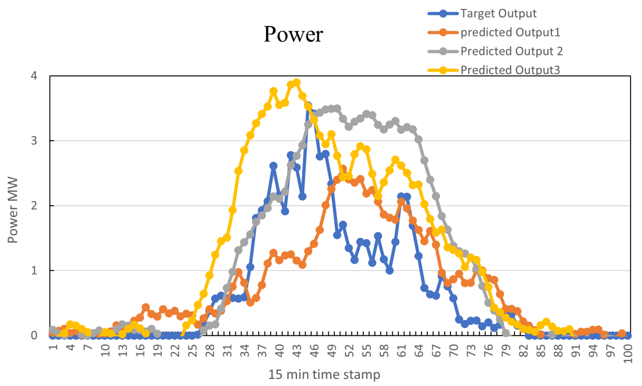

Figure 7. MTSF Prediction

Fig.7 shows the results of the MTSF model. In this model, power is predicted directly one day ahead. The graph represents the last three predictions, which are shown in TABLE VII. This graph provides a visual representation of the model's performance for predicting power one-day ahead and helps to understand the accuracy of the predictions made by the model. Fig.8 displays the results of the MTMF-2 model. TABLE VIII describes test losses for various hyperparameters. This model

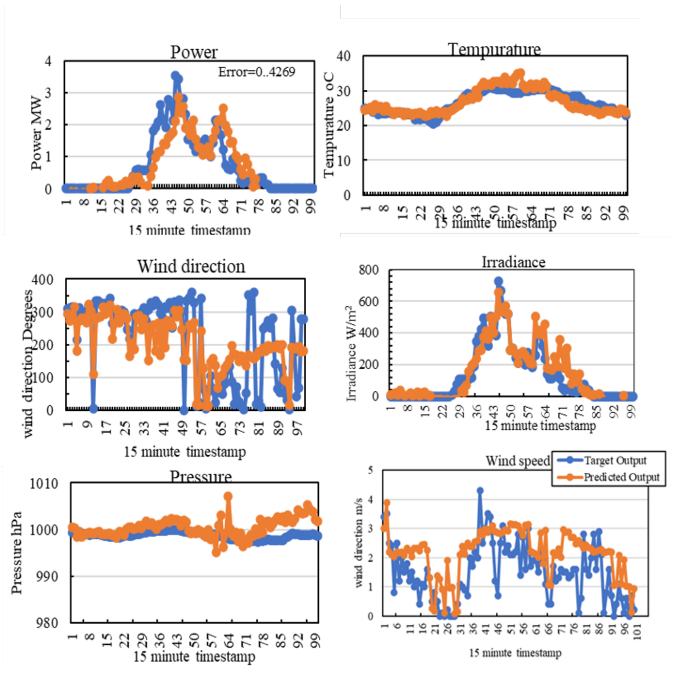

Figure 8. MTMF-2 Prediction

gives a balance between accuracy and practical sense. MTMF-1 and MTSF models also give a practical sense, but MTMF-1 is diverging at starting and MTSF gives more error compared to MTMF-2 model.

VI. Conclusion

The paper proposes four models for power prediction using Convolutional Neural Networks (CNNs) and different data preparation methods. In the STMF model, all features of the next time instant were predicted without incorporating the previous prediction, leading to low error. In the MTMF-1 model, the predicted value of the current time was used for the next prediction, resulting in a higher error. The MTSF model directly predicted the power of the next time instant, which resulted in an even higher error compared to the STMF model. The MTMF-2 model predicted the power output for one day by using past data at the same time. Thus, we conclude that the MTMF-2 model is the best for predicting Photovoltaic (PV) power output compared to the STMF, MTMF-1, and MTSF models.

References

[1] Akhter, Muhammad Naveed, et al. ”Review on forecasting of photovoltaic power generation based on machine learning and metaheuristic techniques.” IET Renewable Power Generation 13.7 (2019): 1009-1023.

[2] N. -A. -. Masood, M. I. Asif, A. M. Alam, S. R. Deeba and T. Aziz, ”Forecasting of Photovoltaic Power Generation: Techniques and Key Factors,” 2019 IEEE Region 10 Symposium (TENSYMP), Kolkata, India, 2019, pp. 457-461, doi: 10.1109/TENSYMP46218.2019.8971337

[3] Xu, W.; Yu, B.; Song, Q.; Weng, L.; Luo, M.; Zhang, F. Economic and Low-Carbon-Oriented Distribution Network Planning Considering the Uncertainties of Photovoltaic Generation and Load Demand to Achieve Their Reliability. Energies 2022, 15, 9639. https://doi.org/10.3390/en15249639

[4] C. -H. Liu, J. -C. Gu and M. -T. Yang, ”A Simplified LSTM Neural Networks for One Day-Ahead Solar Power Forecasting,” in IEEE Access, vol. 9, pp. 17174-17195, 2021, doi: 10.1109/ACCESS.2021.3053638.

[5] Tiechui Yao, Jue Wang, Haoyan Wu, Pei Zhang, Shigang Li, Yangang Wang, Xuebin Chi, Min Shi, A photovoltaic power output dataset: Multi-source photovoltaic power output dataset with Python toolkit, Solar Energy, Volume 230, 2021, Pages 122-130, ISSN 0038-092X, https://doi.org/10.1016/j.solener.2021.09.050.

[6] C. -J. Huang and P. -H. Kuo, ”Multiple-Input Deep Convolutional Neural Network Model for Short-Term Photovoltaic Power Forecasting,” in IEEE Access, vol. 7, pp. 74822-74834, 2019, doi: 10.1109/ACCESS.2019.2921238.

[7] Z. Li, F. Liu, W. Yang, S. Peng and J. Zhou, ”A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects,” in IEEE Transactions on Neural Networks and Learning Systems, vol. 33, no. 12, pp. 6999-7019, Dec. 2022, doi: 10.1109/TNNLS.2021.3084827.

[8] Z. Zhang, ”Improved Adam Optimizer for Deep Neural Networks,” 2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS), Banff, AB, Canada, 2018, pp. 1-2, doi: 10.1109/IWQoS.2018.8624183.