IOT based solar energy prophecy using RNN architecture

V. Vijaya Rama Raju[1,*], *N V* Ganapathi Raju[2], *V* Shailaja[3] and *Sugandha* Padullaparti[4]

[1]Associate Professor, Dept of EEE, GRIET, Hyderabad, India
[2]Professor, Dept of IT, GRIET, Hyderabad, India.
[3]Assistant Professor, Dept of IT, GRIET, Hyderabad, India.
[4]UG student, Dept of IT, GRIET, Hyderabad, India

Abstract. It is the 21st century and scientists say that by the end of this century, resources will be replenished and the only way the future generations can access energy is through renewable resources—those which are inexhaustible. One such source is sunlight, which has a guaranteed stay in the long run. The energy thus given is termed as solar energy. In the present paper it is tried to solve the issue of limited resources and their adverse effects. Since the power generated from solar energy systems is highly variable, due to its dependence on meteorological conditions, an efficient method of usage of this fluctuating but precious energy source has to come in picture. This requires the scope of reliable forecast information as the development of predictive control algorithms for efficient energy management and monitoring for residential grid connected photovoltaic systems. The paper has given an overview of different applications and models for solar irradiance and photovoltaic power prediction, including time series models based on live measured data from rooftop solar power plant located at 17.5203° N, 78.3674° E. For experimentation, data collected over four years from the solar power plant was used in order to the train machine and understand the characteristics of the solar power plant and gives the predicted energy as the result. The use of Deep Learning is done where LSTM is used for the training and keras and tensorflow are used for obtaining the result. The mean square error thus obtained is 0.015.

1 Introduction

Solar energy is pollution free and causes no greenhouse gases to be emitted after installation and there is reduced dependence on foreign oil and fossil fuels. It is a renewable clean power that is available every day of the year, even cloudy days produce some power.

Power generation from solar energy is highly variable due to its dependence on meteorological factors. An efficient use of the fluctuating energy sources requires reliable forecast information for management and operation strategies. Forecasting can be defined as the prediction of some future event or events by analysing the historical data. It spans many areas including business and industry, economics, environmental science and finance.

Solar energy forecasting involves prediction of solar irradiance and photovoltaic power. Irradiance is a measurement of solar power and is defined as the rate at which solar energy falls on a surface. The unit of power is Watt. Photovoltaic power is the electrical current produced by light in solar cells in the solar module. The energy of the light is transformed into electricity in the photovoltaic cells. The electricity can then be used in electrical equipment fed into the electricity network for use by others or stored in a battery.

As the world's population is increasing, so does the demand for energy. To support the ever increasing need to power and to address mounting environmental concerns, many utilities are seeking sustainable power sources to augment traditional ones. Hence there is a need of conserving energy for various activities and the right amount of energy should be used for each activity, and not excessive energy to be used which leads to waste of resources.

An overview of different applications and state of the art models for solar irradiance and photovoltaic power prediction based on measured data from solar power plants was presented in this paper. Deep learning Long Short Term Memory First (LSTM) is used to predict the energy. There is a use of various modules of Python. Data was collected over four years so the machine learns and understands the characteristics of solar power plants. The dataset is taken and the training is done using the above model and the test dataset is internally given for comparison and the output is obtained in the form of a graph.

2 Literature Survey

Ahilan K. et.al [1], have suggested the prediction of solar irradiance using deep learning approaches. Used a deep learning based univariate long short-term memory (LSTM) approach that is introduced to predict the solar irradiance. A univariate LSTM and auto-regressive

[*] vijayram_v@yahoo.com

© The Authors, published by EDP Sciences. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (http://creativecommons.org/licenses/by/4.0/).

integrated moving average (ARIMA) based time series models are compared. Both approaches are evaluated using root mean-square error (RMSE).Their study suggested the better performance of LSTM over ARIMA.

Saad P. et.al.[2], worked on photovoltaic yield prediction using an irradiance forecast model based on multiple neural networks. The proposed irradiance forecast model is based on multiple feed-forward neural networks. Considered a dataset from Stuttgart and found that the global horizontal irradiance forecast has a mean absolute percentage error of 3.4% on a sunny day and 23% on a cloudy day.

Wai M. et al.[3], researched the implementation of solar photovoltaic data monitoring systems. Used many sensors like LM35, Arduino Uno etc. to predict the result of measuring various solar panel parameters.

Kardakos E. et al.[4], proposed two practical methods for electricity generation forecasting of grid-connected PV plants. The first being the application of ARIMA time series and the other being artificial neural network models in short-term forecasting of PV power generation.

Hizam H. et.al.[5], worked on the modelling and prediction of photovoltaic power using artificial neural networks like FFBP(feedforward backpropagation) and GRNN(general regression neural network) showing that FFBP has a better performance than GRNN.

3 Methodology

The methodology of the paper consists of the following steps.

3.1 Remote Monitoring of Photovoltaic Power Plant using IoT device

Monitoring of photovoltaic power plants are vital for reliable operation and high yield of energy for any solar power system. Monitoring can be done in two ways, Manual and Online. Manual monitoring can be done by the operator visiting the plant on a regular basis and collecting the data from the front panel of the inverters. This is the easiest way but tedious of monitoring of an inverter to record the readings from the display. However the number of parameters displayed on the monitor are limited to day energy, instantaneous power, voltage, etc. only rather than the complete history of the system. As the plants are located at the remote places and rooftops of the buildings, manual recordings become tiresome, expensive and most importantly inaccurate involving human errors. For more sophisticated monitoring purposes environmental data – like solar radiation, wind speed along with electrical parameters can also be data logged, stored and analysed later remotely using IoT. This data can be further used to predict the solar plant generation in order to manage the available resources like grid power, Diesel Generators, etc. in a more efficient way. Remote monitoring can be done using various communication technologies like: analog modem, ISDN, GSM etc. For local monitoring USB, RS232, RS485, etc. are the most commonly used technologies. For wireless connection Bluetooth and Wi-Fi communications are used most commonly. Developed an in-house IoT module to collect the data and to send it to the cloud server for remote monitoring.

3.2 IOT Sensor board design for acquisition of solar data

The data has been collected live from GRIET college official Solar Energy through IOT sensors. IoT sensor board shown in the Fig.1 has the following resources on-board to collect the information from solar inverters: Arduino Nano for data processing, ESP8266 Based Wi-Fi Module to push the data to cloud, MAX 485 to data acquisition from the inverter using RS485 communication, External Power supply from Wall Adapter (Included).

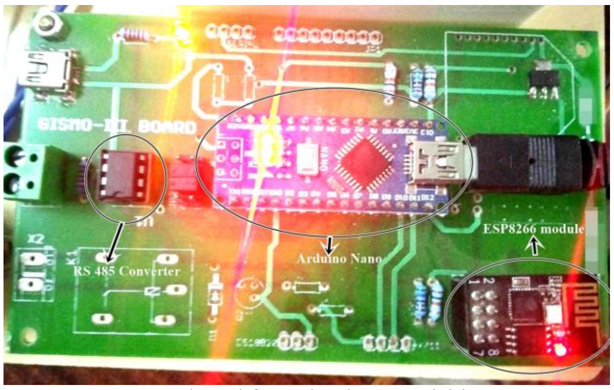


Fig. 1. IOT Sensor board for Solar data acquisition

3.3 Data Acquisition using IoT device

- In order to collect the data from the inverters, IoT devices send a request to the inverter's memory registers to access the information about various parameters like energy, power, AC & DC voltages, currents, irradiation, etc using an RS485 converter.
- After receiving the requested information from the inverter, the IoT device will push this data to the cloud services using ESP8266 module through WiFi.

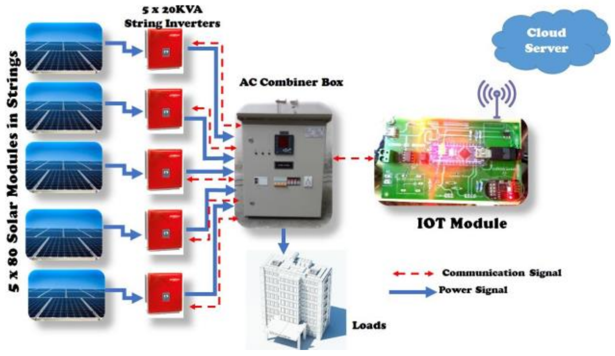
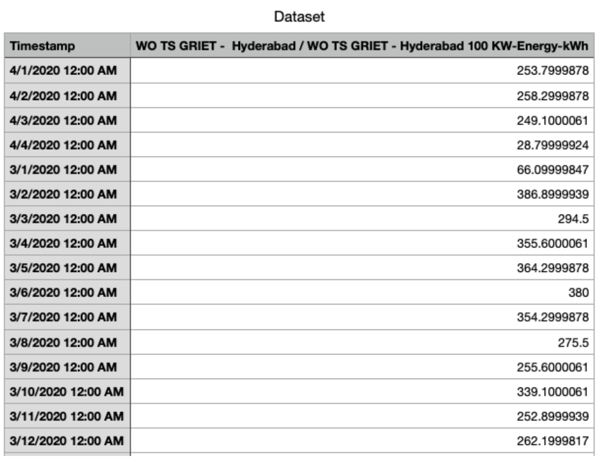


Fig. 2. Architecture diagram

- Mobile applications are developed to access the data from the cloud.
- Historical data in the cloud server was used for prediction applications. The architecture of the paper is shown in Fig 2.

The dataset consists of readings from the solar plant that contain data of over four years ranging from January 2016 to March 2020 along with the first four days in April 2020. There are two columns here-one representing the date and time, and the other representing the solar energy reading from 100 KWp power plant Energy in kWh.


Dataset

Timestamp	WO TS GRIET - Hyderabad / WO TS GRIET - Hyderabad 100 KW-Energy-kWh
4/1/2020 12:00 AM	253.7999878
4/2/2020 12:00 AM	258.2999878
4/3/2020 12:00 AM	249.1000061
4/4/2020 12:00 AM	28.79999924
3/1/2020 12:00 AM	66.09999847
3/2/2020 12:00 AM	386.8999939
3/3/2020 12:00 AM	294.5
3/4/2020 12:00 AM	355.6000061
3/5/2020 12:00 AM	364.2999878
3/6/2020 12:00 AM	380
3/7/2020 12:00 AM	354.2999878
3/8/2020 12:00 AM	275.5
3/9/2020 12:00 AM	255.6000061
3/10/2020 12:00 AM	339.1000061
3/11/2020 12:00 AM	252.8999939
3/12/2020 12:00 AM	262.1999817

Fig 3: The dataset as a csv file

The current research is developed to predict solar energy based on the data read. The large data that is pre-processed is learnt by the machine and is used for prediction.

The following figure shows the import statements as well as the dataset extraction.

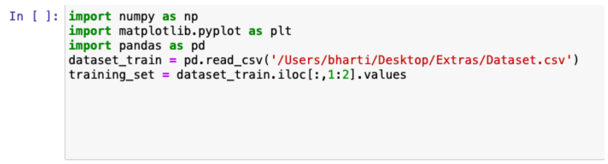

```
In [ ]: import numpy as np
        import matplotlib.pyplot as plt
        import pandas as pd
        dataset_train = pd.read_csv('/Users/bharti/Desktop/Extras/Dataset.csv')
        training_set = dataset_train.iloc[:,1:2].values
```

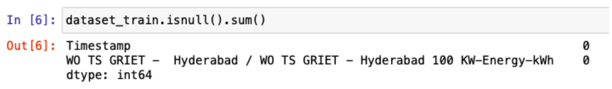
Fig 4: Code snippet showing import statements and reading data

3.4 Data pre-processing

Data received from the IOT sensors will be pre-processed using filters by finding, cleaning and preparing the appropriate data.

3.4.1 Removing missing values

First, we check if there is any missing data. If so, we can choose either the mean, median or mode in place of the missing value. Depending on our wish. This is called data imputation. The function then used is fillna().


```
In [6]: dataset_train.isnull().sum()

Out[6]: Timestamp                                                          0
        WO TS GRIET -  Hyderabad / WO TS GRIET - Hyderabad 100 KW-Energy-kWh    0
        dtype: int64
```

Fig 5: Code snippet that shows how we check for presence of null data

3.4.2. Standardization of data/Feature Scaling

Using MinMaxScalar the data has been normalized between 0 to 1. Once the pre-processing step is completed, prepare the training set from the dataset.

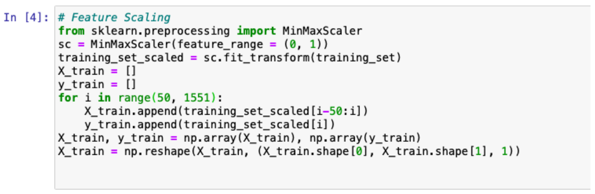
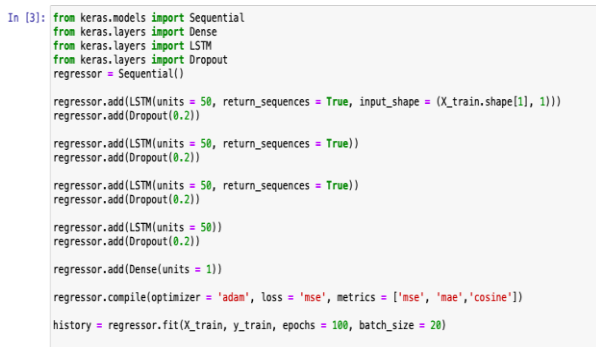

```
In [4]: # Feature Scaling
        from sklearn.preprocessing import MinMaxScaler
        sc = MinMaxScaler(feature_range = (0, 1))
        training_set_scaled = sc.fit_transform(training_set)
        X_train = []
        y_train = []
        for i in range(50, 1551):
            X_train.append(training_set_scaled[i-50:i])
            y_train.append(training_set_scaled[i])
        X_train, y_train = np.array(X_train), np.array(y_train)
        X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))
```

Fig 6: Code snippet explaining how training set is obtained along with data normalization

3.5 Choosing a model

We choose the model for the prediction using Time Series analysis with Long Short Term Memory (LSTM) algorithm of CNN Model. Since the dataset contains only two columns viz., timestamp and the solar energy values.


```
In [3]: from keras.models import Sequential
        from keras.layers import Dense
        from keras.layers import LSTM
        from keras.layers import Dropout
        regressor = Sequential()

        regressor.add(LSTM(units = 50, return_sequences = True, input_shape = (X_train.shape[1], 1)))
        regressor.add(Dropout(0.2))

        regressor.add(LSTM(units = 50, return_sequences = True))
        regressor.add(Dropout(0.2))

        regressor.add(LSTM(units = 50, return_sequences = True))
        regressor.add(Dropout(0.2))

        regressor.add(LSTM(units = 50))
        regressor.add(Dropout(0.2))

        regressor.add(Dense(units = 1))

        regressor.compile(optimizer = 'adam', loss = 'mse', metrics = ['mse', 'mae','cosine'])

        history = regressor.fit(X_train, y_train, epochs = 100, batch_size = 20)
```

Fig 7: Code snippet showing steps involved for training a model

3.6 Training the model

The dataset is to be trained to predict the future so keras layers along with Dense and Dropout were used, which are the core layers from CNN algorithm for Deep Learning. To build a network, Sequential is the model that has been imported from keras. Train the model using fit().

3.7 Evaluating the model

As the training is done in the previous section, evaluating the model to be done in order of 70/30 where 70% is for training while 30% is for testing.

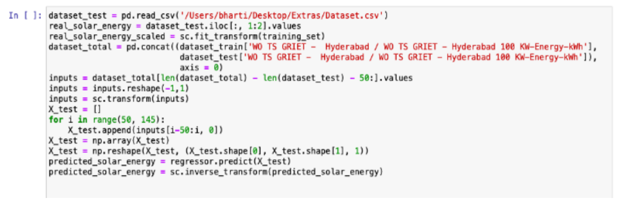

```
In [ ]: dataset_test = pd.read_csv('/Users/bharti/Desktop/Extras/Dataset.csv')
        real_solar_energy = dataset_test.iloc[:, 1:2].values
        real_solar_energy_scaled = sc.fit_transform(training_set)
        dataset_total = pd.concat((dataset_train['WO TS GRIET -  Hyderabad / WO TS GRIET - Hyderabad 100 KW-Energy-kWh'],
                                   dataset_test['WO TS GRIET -  Hyderabad / WO TS GRIET - Hyderabad 100 KW-Energy-kWh']),
                                   axis = 0)
        inputs = dataset_total[len(dataset_total) - len(dataset_test) - 50:].values
        inputs = inputs.reshape(-1,1)
        inputs = sc.transform(inputs)
        X_test = []
        for i in range(50, 145):
            X_test.append(inputs[i-50:i, 0])
        X_test = np.array(X_test)
        X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
        predicted_solar_energy = regressor.predict(X_test)
        predicted_solar_energy = sc.inverse_transform(predicted_solar_energy)
```

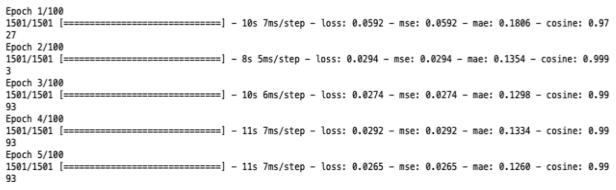
Fig 8: Code snippet for evaluation of model with test dataset

4 Result Analysis

The performance of the model can be evaluated by taking into consideration using the following factors: (1) Error calculated and (2) Visual analysis in the form of a plot

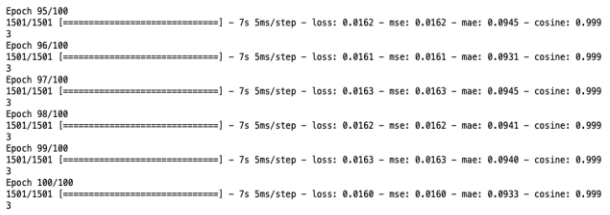
4.1 Error Calculated

The error obtained in the output is reduced from 0.04 in the first epoch to 0.015 in the last epoch. This is clearly shown in the figures below.



```
Epoch 1/100
1501/1501 [==============================] - 10s 7ms/step - loss: 0.0592 - mse: 0.0592 - mae: 0.1806 - cosine: 0.97
27
Epoch 2/100
1501/1501 [==============================] - 8s 5ms/step - loss: 0.0294 - mse: 0.0294 - mae: 0.1354 - cosine: 0.999
3
Epoch 3/100
1501/1501 [==============================] - 10s 6ms/step - loss: 0.0274 - mse: 0.0274 - mae: 0.1298 - cosine: 0.99
93
Epoch 4/100
1501/1501 [==============================] - 11s 7ms/step - loss: 0.0292 - mse: 0.0292 - mae: 0.1334 - cosine: 0.99
93
Epoch 5/100
1501/1501 [==============================] - 11s 7ms/step - loss: 0.0265 - mse: 0.0265 - mae: 0.1260 - cosine: 0.99
93
```

Fig 9: The initial epochs generated



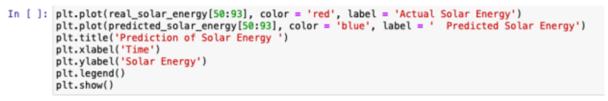
```
Epoch 95/100
1501/1501 [==============================] - 7s 5ms/step - loss: 0.0162 - mse: 0.0162 - mae: 0.0945 - cosine: 0.999
3
Epoch 96/100
1501/1501 [==============================] - 7s 5ms/step - loss: 0.0161 - mse: 0.0161 - mae: 0.0931 - cosine: 0.999
3
Epoch 97/100
1501/1501 [==============================] - 7s 5ms/step - loss: 0.0163 - mse: 0.0163 - mae: 0.0945 - cosine: 0.999
3
Epoch 98/100
1501/1501 [==============================] - 7s 5ms/step - loss: 0.0162 - mse: 0.0162 - mae: 0.0941 - cosine: 0.999
3
Epoch 99/100
1501/1501 [==============================] - 7s 5ms/step - loss: 0.0163 - mse: 0.0163 - mae: 0.0940 - cosine: 0.999
3
Epoch 100/100
1501/1501 [==============================] - 7s 5ms/step - loss: 0.0160 - mse: 0.0160 - mae: 0.0933 - cosine: 0.999
3
```

Fig 10: The final epochs generated

4.2 Visual Analysis

The best way to do an analysis using plots, is to use matplotlib, which can be imported as a module in python. In the present paper, plot two things viz., prediction and error analysis.

1. To get a plot that shows the actual values and predicted values from the test dataset created, the following code gives a clear understanding.



```
In [ ]: plt.plot(real_solar_energy[50:93], color = 'red', label = 'Actual Solar Energy')
        plt.plot(predicted_solar_energy[50:93], color = 'blue', label = '  Predicted Solar Energy')
        plt.title('Prediction of Solar Energy ')
        plt.xlabel('Time')
        plt.ylabel('Solar Energy')
        plt.legend()
        plt.show()
```

Fig 11: Code snippet showing the steps involved for obtaining a plot

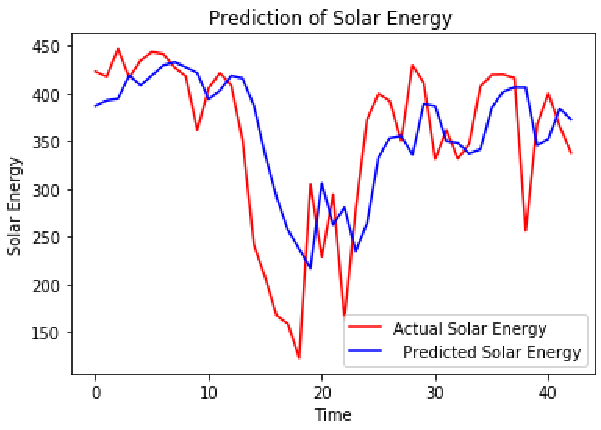
- Matplotlib is imported as plot as shown in the first step. The visualisation is done with a plot with appropriate labels and colors. The plot thus obtained is as follows:

Fig 12: Prediction of Solar Energy

- As we can see in the figure, there is a significant similarity between the Actual Solar Energy and Predicted Solar Energy.
- The X-axis talks about the time in months (approximately 45 months) and the Y-axis talks about the solar energy generated by the 100 KWp solar power plant Energy in kWh scale.

2. Next, plotted the various errors calculated in the epochs. The paper considers, three errors as follows

- Mean Squared Error (MSE): It measures the average of error squares i.e. the average squared



difference between the estimated values and true value.

- Mean Absolute Error (MAE): It is a quantity used to measure how close forecasts or predictions are to the eventual outcomes.
- Cosine Proximity: It is a measure of similarity between two vectors. Here, the predicted and actual values of the test dataset.

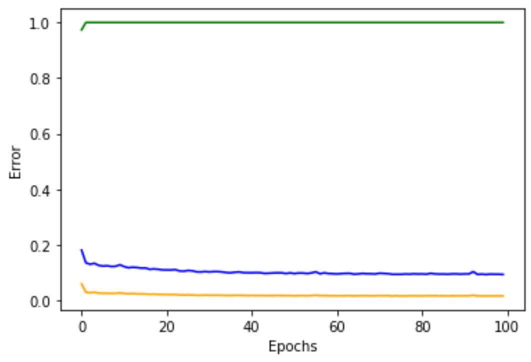


Fig 13: Error plots for MSE, MAE, and Cosine Proximity

From the above Fig.13, plot characteristics can be observed as follows:

- X-axis has the number of epochs that is 100 epochs as shown in the above figures.
- Y-axis talks about the errors.
- The green line represents the cosine proximity. The value is 0.99 approximately and is constant. This is close to 1, which means the two vectors are almost identical.
- The blue line represents mean absolute error which lies within range of 0.2
- The orange line represents mean squared error which lies within range of 0.1

As these errors are low, we can conclude that our model is good.

5 Conclusion

The paper thus concludes that there is immense need for protecting the resources we have currently for future use. Implementing renewable energy technologies is one recommended way of reducing the environmental impact. Solar energy is a non-polluting and effective way of doing this. Artificial Intelligence is a fast growing language in the industry today. The sub module of this is Machine Learning that consists of many algorithms and models which can be used to understand the various situations or problems in real life that can be implemented with programming. Python is the latest programming language and the most sought after subject. So there is a lot in store for those who use ML and here Deep Learning-that is Machine Learning is used, but that has a lot of datasets. The scope of this paper can be extended to various areas including domestic customers where residents have ample idea about the solar energy they are going to get for a short future time so that they can conserve the energy accordingly. The electricity can thus be efficiently used. If such systems are installed more in number, the pollution can reduce drastically, along with saving a lot of non-renewable resources for the future. Future is the new present. Thus, can be improved in many ways such as implementation of better models for improved results. Also, cheaper ways of installing sensors can be quite helpful as we require that all households access this facility. Development implies a tech-savvy country. The future lies in technology.

Acknowledgment

The authors would like to thank GRIET management for their support in carrying out this research work at the institute using the solar plant data.

References

1. Sharika, Wathmini, et al. "Long-term Solar Irradiance Forecasting Approaches-A Comparative Study". *IEEE International Conference on Information and Automation for Sustainability*, (2018).
2. Durrani, Saad Parvaiz, et al. "Photovoltaic yield prediction using an irradiance forecast model based on multiple neural networks." *Journal of Modern Power Systems and Clean Energy* **6.2** (2018): 255-267.
3. Yang, Handa, et al. "Solar irradiance forecasting using a ground-based sky imager developed at UC San Diego." *Solar Energy* 103 (2014): 502-524.
4. Kardakos, Evaggelos G., et al. "Application of time series and artificial neural network models in short-term forecasting of PV power generation." *48th International Universities' Power Engineering Conference (UPEC)*. IEEE, (2013).
5. Saberian, Aminmohammad, et al. "Modelling and prediction of photovoltaic power output using artificial neural networks." *International journal of Photoenergy* (2014).
6. V. Vijaya Rama Raju, Divya Mereddy, "Smart Dual Axes Solar Tracking System", *2015 IEEE International Conference on Energy Systems and Applications (ICESA 2015)* Nov, 2015.
7. Purohit, Rajesh, et al. "Optimization of electric discharge machining of M2 tool steel using grey relational analysis." *Materials Today: Proceedings* **2.4-5** (2015): 3378-3387.
8. J Praveen, V Vijaya Rama Raju, "Materials for Optimizing Efficiencies of Solar Photovoltaic Panels", *Elsevier, Science Direct Materials Today: Proceedings* **4** (2017) 5233–5238
9. Lakshmi, K. Prasanna, and C. R. K. Reddy. "A survey on different trends in data streams." *International Conference on Networking and Information Technology*. IEEE, 2010.