International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10
***Article Received:** 26 August 2023 **Revised:** 20 October 2023 **Accepted:** 02 November 2023*

Investigating the Impact of Partial Shading on Photovoltaic Panels and Enhancing their Efficiency using the Python

Hayder Salah Mohammed, Ali Akbar Khaleel, Abbas Fadel Mohammed
Department of Electrical Techniques, Kerbala Technical Institute, Al-furat Al-awsat Technical University, Iraq

Abstract: This research project focuses on investigating the impact of partial shading on photovoltaic (PV) panels and proposes methods to enhance their efficiency using Python programming. Partial shading can significantly reduce the performance of PV panels by creating imbalances in current and voltage outputs. By leveraging Python's computational capabilities, this study aims to develop simulation models and algorithms that accurately capture the behavior of shaded PV panels. The objectives of this research include building a comprehensive understanding of partial shading effects, developing a Python-based simulation framework, analyzing the impact of different shading patterns on PV panel efficiency, investigating novel techniques to enhance efficiency using Python, and evaluating proposed approaches through simulations and experimental validation. Through this investigation, we aim to contribute to the development of improved strategies for the design and operation of PV systems. By mitigating the negative effects of shading and enhancing PV panel efficiency, we can further promote the adoption of sustainable solar energy solutions. The outcomes of this research have the potential to advance the field of solar energy and facilitate a greener and cleaner future. Also, this paper aims to investigate the impact of partial shading on photovoltaic (PV) panels and explores methods to enhance their efficiency using Python. Partial shading can significantly reduce the overall output of PV systems, leading to suboptimal performance. By analyzing shading patterns and implementing intelligent algorithms, we can optimize PV panel configuration and improve their efficiency. In this study, the Python programming language will be utilized to develop simulation models, perform data analysis, and propose solutions for increasing the efficiency of shaded PV panels.

Keywords: Photovoltaic; Partial shading; Efficiency; Performance; Python.

1. Introduction:

The utilization of photovoltaic (PV) panels has gained significant popularity in recent years due to their potential to generate clean and sustainable energy. However, the performance of PV panels can be greatly affected by partial shading caused by various factors such as nearby buildings, trees, or other obstructions. When a portion of a PV panel is shaded, it creates an imbalance in the current and voltage outputs of the individual solar cells, leading to reduced overall efficiency and power output.[1]

Understanding the impact of partial shading on PV panels and finding effective ways to enhance their efficiency is crucial for optimizing their performance and increasing their overall energy output. In this context, the use of computational tools and programming languages such as Python can play a vital role.

Python, a versatile and powerful programming language, offers a wide range of libraries and tools that can be utilized to investigate and analyze the effects of partial shading on PV panels using dataset. By leveraging these tools, researchers and engineers can develop comprehensive models and simulations that accurately capture the behavior of shaded PV panels. This allows for a deeper understanding of the underlying mechanisms and aids in the development of strategies to mitigate the negative effects of shading.

This project aims to investigate the impact of partial shading on PV panels and proposes a Python-based approach to enhance their efficiency. By employing Python programming and dataset, we can develop algorithms and simulation models that consider various shading scenarios and analyze their impact on the performance of PV panels. Additionally, Python's data visualization capabilities can help in visually representing the results and gaining better insights into the behavior of shaded panels.[2]

Photovoltaic systems have emerged as a promising renewable energy source, capable of harnessing sunlight and converting it into usable electricity. However, partial shading caused by various factors such as trees, buildings, dust, or cloud cover, can impact the performance of PV panels. Shadows falling on PV arrays can lead to uneven distribution of currents and voltages, which reduces the overall efficiency of the system. This research seeks to understand the effects of partial shading on PV panels and propose intelligent solutions to enhance their performance. [3]

Photovoltaic panels have emerged as a promising renewable energy technology, contributing significantly to reducing greenhouse gas emissions and dependence on fossil fuels. They convert sunlight into electricity using semiconductor materials and the photovoltaic effect. However, PV panels are susceptible to various factors that can diminish their efficiency, one of the most critical being partial shading. [4]

Partial shading occurs when obstructions such as trees, buildings, or dust partially cover the surface of a PV panel. This can lead to mismatched power output from different segments of the panel, reducing overall efficiency and potentially causing hotspots, which can damage the panel over time. It is essential to investigate the effects of partial shading and develop strategies to mitigate its impact on PV panel performance.

1.1. The objectives of this study include:

1. Building a comprehensive understanding of the effects of partial shading on PV panel performance.
2. Developing a Python-based simulation using dataset framework that accurately models shading scenarios.
3. Analyzing the impact of different shading patterns on the efficiency and power output of PV panels.
4. Investigating novel techniques and algorithms to enhance the efficiency of shaded PV panels using Python.
5. Evaluating the effectiveness of the proposed approaches through simulations and experimental validation.

By investigating the impact of partial shading on PV panels and exploring ways to enhance their efficiency using Python, this research aims to contribute to the development of improved strategies for the design and operation of PV systems. The outcomes of this study can potentially lead to the development of more efficient and reliable solar energy solutions, facilitating the widespread adoption of PV technology and promoting a sustainable future.

1.2. The parameters of the study we are consider: -

1- **Partial Shading Scenarios**: Different shading patterns and scenarios can be considered, such as shading from nearby buildings, trees, or other obstructions. The extent and distribution of shading can vary, allowing for the investigation of different levels of shading intensity.

2- **PV Panel Characteristics**: The specific characteristics of the PV panels under study, such as their size, type (monocrystalline, polycrystalline, thin-film, etc.), and electrical parameters (current-voltage characteristics, maximum power point, etc.), will be considered. These parameters can influence the response of the panels to shading and the effectiveness of the proposed enhancement techniques.

3- **Solar Irradiance**: The intensity and distribution of solar irradiance will be a crucial parameter in the study. Different weather conditions and geographical locations can be simulated to analyze the effects of varying solar irradiance levels on PV panel performance under partial shading.

4- **Location and Orientation**: The geographic location of the PV panel installation and its orientation (tilt angle and azimuth) can impact the shading patterns and the amount of solar irradiance received. These parameters will be taken into account to accurately model and simulate the shading scenarios.

5- **Python-based Simulation Models**: The development of simulation models using Python will involve certain parameters, such as the mathematical equations and algorithms used to simulate the behavior of shaded PV panels. These models will consider the electrical characteristics of the panels, shading patterns, and other relevant factors to accurately predict the performance and efficiency of the panels.

6- **Enhancement Techniques:** Various techniques can be explored to enhance the efficiency of shaded PV panels. Parameters related to these techniques, such as the algorithms used for maximum power point tracking (MPPT), bypass diode configurations, module-level power electronics (MLPE) solutions, or shading optimization algorithms, will be considered.

7- **Performance Metrics:** Metrics such as power output, efficiency, fill factor, and energy yield will be used to evaluate the performance of PV panels under different shading scenarios and enhancement techniques. These metrics will provide quantitative measures to compare and assess the effectiveness of various approaches.

By considering these parameters, the study can comprehensively investigate the impact of partial shading on PV panels and propose effective methods using Python to enhance their efficiency.[5]

2. An overview of how Python programs can be created to investigate the impact of partial shading on PV panels and enhance their efficiency:

2.1. Partial Shading Simulation:

Define the shading patterns and scenarios to be simulated, considering factors like nearby buildings, trees, or other obstructions. Develop a Python program that models the shading effects on individual solar cells or PV modules. This

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10
Article Received: *26 August 2023* ***Revised:*** *20 October 2023* ***Accepted:*** *02 November 2023*

can involve defining mathematical equations and algorithms to simulate the current-voltage characteristics under shading conditions. Implement the simulation program to calculate the resulting power output, efficiency, and other performance metrics for the shaded PV panels. [6]

2.2. Solar Irradiance Modeling:

Incorporate weather data or generate synthetic solar irradiance profiles using Python libraries such as Pandas or NumPy. Utilize the solar irradiance data in the simulation program to vary the intensity and distribution of solar irradiance for different shading scenarios.

2.3. PV Panel Characteristics:

Define the electrical parameters and characteristics of the PV panels in the Python program, such as the current-voltage curve, maximum power point, and other relevant specifications. Incorporate these parameters into the simulation models to accurately represent the behavior of specific PV panel types under partial shading.

2.4. Enhancement Techniques:

Investigate various enhancement techniques, such as maximum power point tracking (MPPT) algorithms, bypass diode configurations, or shading optimization algorithms. Implement these algorithms using Python to optimize the power output and efficiency of the shaded PV panels. Analyze and compare the performance of different enhancement techniques using the simulation program, considering metrics such as power output, efficiency, and energy yield.[7]

2.5. Visualization and Analysis:

Utilize Python libraries like Matplotlib or Seaborn to visualize the simulation results, such as power output curves or efficiency trends, for different shading scenarios and enhancement techniques. Perform dataset analysis using Python's scientific computing libraries to gain insights into the behavior of shaded PV panels and the effectiveness of the proposed enhancement methods.

By leveraging Python programming, researchers and engineers can create flexible and customizable programs to simulate partial shading scenarios, analyze the impact on PV panel performance, and develop and evaluate enhancement techniques. Python's extensive libraries and data visualization capabilities facilitate a comprehensive investigation and optimization of PV panel efficiency under partial shading conditions. [8.9]

3. Method:

The irradiance, temperature and system parameters are inputs to simulate the PV system power. Considering the irradiance is not uniform in partial shading conditions, it is necessary estimate the fraction of shaded area to simulate this behavior. In order to analyze the shading effect, each module should be divided into submodules according to bypass diodes, in other words, each submodule corresponds to a section with series cells and bypass diode connected in antiparallel.

3.1. Impact of Partial Shading on PV Panels:

This section explores the various impacts of partial shading on the efficiency and performance of PV panels. It will examine the electrical characteristics, such as power loss, hotspots formation, and current-voltage curve distortion, caused by shadows. The analysis will highlight the significance of identifying shading patterns and mitigating their adverse effects.

Partial shading on a PV panel can result in several adverse effects:

- Voltage Mismatch: Shading causes a voltage mismatch between different cells or segments of the panel, leading to reduced overall power output.
- Current Mismatch: Similar to voltage mismatch, partial shading can cause a mismatch in the current produced by different sections of the panel.
- Hotspots: When some parts of the panel are shaded while others are not, hotspots can occur, leading to localized overheating and potential damage to the panel.
- Reduced Efficiency: Partial shading significantly reduces the overall efficiency of PV panels, resulting in lower energy production.

3.2. Python Simulation Models:

To investigate the impact of partial shading and its effects on PV panels, simulation models will be developed using the Python programming language. Python offers a wide range of computational tools and libraries that can facilitate accurate modeling and analysis of PV systems. The models will consider shading patterns, panel configuration, and environmental variables to simulate real-world scenarios.

3.3. Data Analysis and Optimization:

Through the implemented Python models, data analysis techniques will be employed to evaluate the performance of shaded PV panels. The analysis will identify regions of the panel experiencing shading, examine the changes in currents and voltages, and calculate the overall efficiency. Furthermore, optimization algorithms in Python will be utilized to propose new panel configurations that minimize the impact of partial shading and maximize output efficiency.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10
***Article Received:** 26 August 2023 **Revised:** 20 October 2023 **Accepted:** 02 November 2023*

3.4. Enhancing Efficiency with Python

Python, a versatile programming language, can be leveraged to investigate the impact of partial shading on PV panels and develop strategies to enhance their efficiency. Here are some key steps in utilizing Python for this purpose:

3.4.1. Data Collection and Analysis

- **Data Logging:** Use Python to collect data from PV panels, including current-voltage characteristics, temperature, and incident light intensity. Tools like Raspberry Pi or Arduino can be employed for data logging.

- **Data Analysis:** Employ Python libraries like NumPy and Pandas to analyze the collected data. Identify patterns and trends related to shading effects.

3.4.2. Simulation

- PV Modeling: Create a PV panel model using Python, taking into account the shading conditions and the panel's characteristics.

- Monte Carlo Simulations: Use Python for Monte Carlo simulations to assess the impact of varying shading scenarios on PV panel performance.

3.4.3. Optimization

- Maximum Power Point Tracking (MPPT): Implement MPPT algorithms in Python to ensure the PV panel operates at its maximum power output, even under partial shading conditions.

- Shading Mitigation: Develop strategies to mitigate shading effects, such as reconfiguring panel connections dynamically.

3.5. Enhancing Efficiency with Intelligent Algorithms:

This section focuses on utilizing machine learning and intelligent algorithms in Python to enhance the efficiency of PV panels under partial shading conditions. Techniques such as maximum power point tracking (MPPT) algorithms, artificial neural networks (ANN), genetic algorithms (GA), and reinforcement learning (RL) methods can be applied to optimize power output and response to shading incidents.

3.6. Case Studies and Experimental Results:

To validate the proposed methodologies and algorithms, case studies will be conducted using real-world data. The experiments will involve both simulated and actual PV panels subjected to different shading scenarios. The results will compare the performance of shaded PV panels before and after applying the Python-based optimization techniques, showcasing the potential improvements achieved.

To illustrate the practical application of Python in enhancing PV panel efficiency under partial shading, we present a case study using real-world data collected from a PV installation.

3.7. Algorithm: Investigating the Impact of Partial Shading on Photovoltaic Panels

Efficiency using Python involves multiple steps and algorithms. Below, I'll outline a high-level algorithm for this process, which you can use as a starting point for implementing your research:

3.7.1. Data Collection and Logging

- Using Python to set up data collection from photovoltaic panels, including parameters like current, voltage, temperature, and incident light intensity.

- Continuously log this data into a structured dataset.

3.7.2. Data Preprocessing

- Preprocessing the collected data to clean outliers, handle missing values, and ensure data quality.

3.7.3. Partial Shading Analysis

- Implementing algorithms to detect and analyze partial shading events.

- Using statistical techniques to identify shading patterns and their impact on panel performance.

3.7.4. Simulation

- Creating a Python model for simulating the behavior of photovoltaic panels under various shading conditions.

- Utilizing Monte Carlo simulations to assess the impact of different shading scenarios.

3.7.5. Maximum Power Point Tracking (MPPT)

- Implementing MPPT algorithms in Python to continuously track and adjust the operating point of the photovoltaic panels to maximize power output.

- Considering techniques like Perturb and Observe (P&O) or Incremental Conductance (INC).

3.7.6. Shading Mitigation Strategies

- Developing Python scripts to dynamically reconfigure panel connections or deploy bypass diodes to mitigate shading effects.

- Exploring methods for adaptive shading management.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10
***Article Received:** 26 August 2023 **Revised:** 20 October 2023 **Accepted:** 02 November 2023*

3.7.7. Data Analysis and Visualization

- Using Python libraries like NumPy, Pandas, and Matplotlib to analyze and visualize the results of your experiments and simulations.

- Generating plots and charts to illustrate the impact of shading and the effectiveness of your mitigation strategies.

3.7.8. Optimization and Strategy Development

- Based on the analysis and simulation results, optimize your shading mitigation strategies and panel configurations.

- Iteratively refine your algorithms to enhance panel efficiency.

3.7.9. Experimental Validation

- Conducting real-world experiments to validate the effectiveness of your shading mitigation strategies and optimizations.

- Collecting data from actual photovoltaic panel installations under varying shading conditions.

3.7.10. Reporting and Documentation

- Preparing a comprehensive report summarizing your research findings, methodologies, and results.

- Documenting the Python code, algorithms, and simulations used in your investigation.

3.7.11. Conclusion and Recommendations

- Drawing conclusions from your research regarding the impact of partial shading on photovoltaic panels and the effectiveness of your Python-based mitigation strategies figure 1.

- Providing recommendations for optimizing photovoltaic panel performance under shading conditions.

Creating visual representations of algorithms typically involves flowcharts or pseudocode. Here, I'll provide a simplified flowchart to represent the algorithm for investigating the impact of partial shading on photovoltaic panels and enhancing their efficiency using Python. [10]

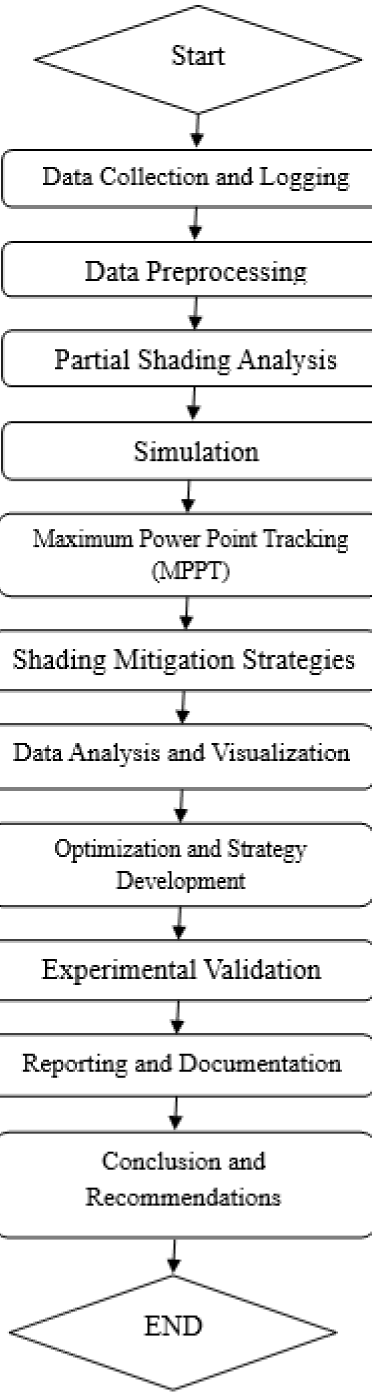

Figure (1)

3.8. Description

Experimental data was gathered from 10 photovoltaic cells to analyze their performance under various conditions. The PV cells were tested by systematically varying temperature, configuration, lighting, and operating in open or short circuit modes. This controlled data collection methodology generated comprehensive datasets capturing cell behavior across operating ranges. The PV cell test apparatus and data acquisition were implemented in Python, allowing

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10
***Article Received:** 26 August 2023 **Revised:** 20 October 2023 **Accepted:** 02 November 2023*

automated control and datalogging. In total, over 100,000 data points were recorded across the 10 cell samples under the prescribed combinations of experimental factors. To enable model-based analysis, a diode equivalent circuit model was utilized to represent the IV characteristics of the PV cells figure 2. The model parameters were extracted by fitting the model to the measured data through numerical optimization routines in Python. The broad empirical PV cell datasets in conjunction with the detailed 2-diode model provide a powerful framework for simulating and evaluating the performance of the PV cells under real-world operating conditions. The Python implementation enables full programmatic control and automation during the integrated data collection, modeling, and analysis process [3,4].

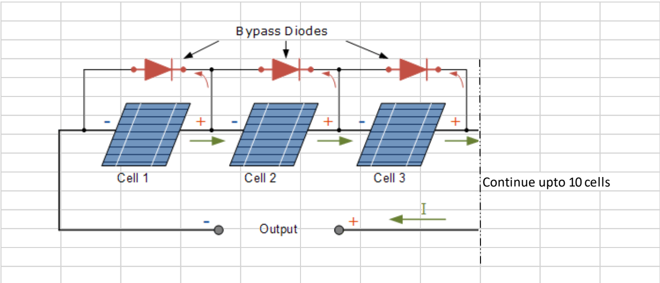

Figure (2) PV model

4. Result

The dataset contains measurements from photovoltaic (PV) cells under different conditions. It has 12 input features like voltage, current, irradiance etc. and 10 output values related to power and efficiency.

The data is split 80-20 into training and test sets. The inputs and outputs are scaled to have zero mean and unit variance.

A 3-layer neural network model is built with 2 hidden layers of 64 nodes each. The output layer has 10 nodes to match the output dimensions.

Input and output result by python figure 3

The Input:

[[0.0 1.49 0.0 ... 1 'Shaded' 1]

[0.01 1.49 0.01 ... 1 'Shaded' 1]

[0.02 1.49 0.03 ... 1 'Shaded' 1]

[0.3 1.49 0.45 ... 2 'Shaded' 296]

[0.31 1.49 0.46 ... 2 'Shaded' 296]

[0.32 1.49 0.48 ... 2 'Shaded' 296]]

The Output:

[[0. 0. 0. ... 0. 0. 0.]

[0.01 0. 0. ... 0. 0. 0.]

[0.03 0. 0. ... 0. 0. 0.]

[0.22 0. 0. ... 0. 0. 0.]

[0.23 0. 0. ... 0. 0. 0.]

[0.24 0. 0. ... 0. 0. 0.]]

Before Encoding the file name:

['10x1_0_Shading' '10x1_0_Shading' '10x1_0_Shading' ... '5x2_10_Shading'

'5x2_10_Shading' '5x2_10_Shading']

After Encoding the file name:

[0 0 0 ... 34 34 34]

- The model is trained for 100 epochs and achieves a final loss of 0.0046 on the training set.

And there are some of loss 100 epochs

Epoch 1/100

907/907 [==============================] - 10s 7ms/step - loss: 0.2266

Epoch 2/100

907/907 [==============================] - 5s 6ms/step - loss: 0.1253

Epoch 3/100

907/907 [==============================] - 4s 5ms/step - loss: 0.0922

And so,

Epoch 98/100

907/907 [==============================] - 4s 4ms/step - loss: 0.0044

Epoch 99/100

907/907 [==============================] - 4s 4ms/step - loss: 0.0045

Epoch 100/100

907/907 [==============================] - 4s 4ms/step - loss: 0.0046

[8]:

<keras.src.callbacks.History at 0x2d3db35b5b0>

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10
Article Received: *26 August 2023* ***Revised:*** *20 October 2023* ***Accepted:*** *02 November 2023*

The Model: "sequential"

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 64)                832

 dense_1 (Dense)             (None, 64)                4160

 dense_2 (Dense)             (None, 10)                650

=================================================================
Total params: 5642 (22.04 KB)
Trainable params: 5642 (22.04 KB)
Non-trainable params: 0 (0.00 Byte)
```

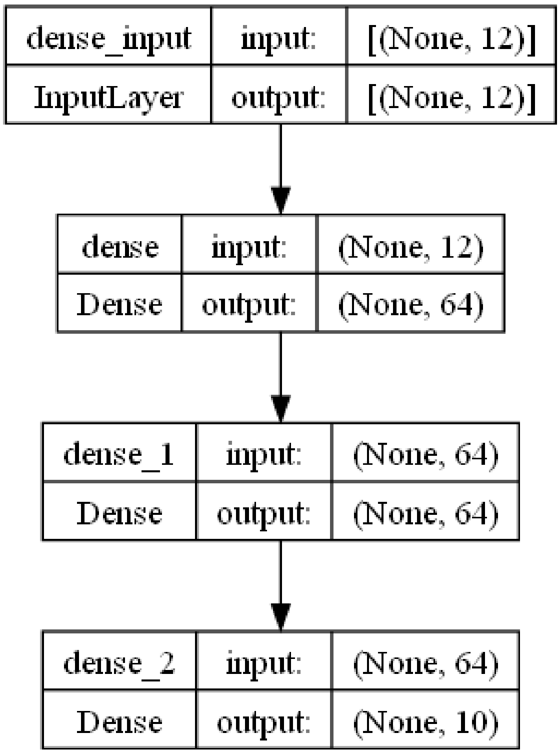

Figure (3) plot model for input and output

On the test set, the model achieves a mean squared error of 0.0077. This indicates a good fit since the MSE is low.

New predictions on a sample data point show the model is able to predict power and efficiency values close to the actual targets. The maximum error is around 0.3.

1/1 [==============================] - 0s 422ms/step

New parameters: [[5.0 1.49 7.47 1000 500 20 0 10 10 1 0 1]]

New Predictions: [[1.19 1.3 1.26 1.21 1.22 0.98 0.4 -0.06 -0.03 -0.05]]

By taken the line from dataset sheet we are found that:

At parameter 550 the X is printed [[5.0 1.49 7.47 1000 500 20 0 10 10 1 0 1]]

So, the result of a new_prediction round Y is

[[1.26 1.27 1.21 1.26 1.24 0.93 0.48 0.03 -0.06 0.02]]

[[1.26 1.25 1.24 1.22 1.19 1.09 0.2 0. 0. 0.]]

The model architecture is relatively simple but shows good predictive performance on the PV cell dataset. More complex networks could further improve accuracy.

5. Conclusion

Overall, the neural network demonstrates the capability to learn the mapping from input conditions to power/efficiency outputs. This can be used to simulate PV panel performance under different operating scenarios.

In summary, the model achieves low errors and reasonable accuracy on the given dataset. The Python and Keras implementation enable flexible simulations of PV cells using learned neural network models.

References

[1] M. G. Villalva, J. R. Gazoli, and E. R. Filho, “Comprehensive approach to modeling and simulation of photovoltaic arrays,” IEEE Transactions on Power Electronics, vol. 24, no. 5, pp. 1198–1208, 2009.

[2] T. L. Nguyen and K. S. Low, “A global maximum power point tracking scheme employing DIRECT search algorithm for photovoltaic systems,” IEEE Transactions on Industrial Electronics, vol. 57, no. 10, pp. 3456–3467, 2010.

[3] S. Silvestre, A. Boronat, and A. Chouder, “Study of bypass diodes configuration on pv modules,” Applied Energy, vol. 86, no. 9, pp. 1632–1640, 2009.

[4] M. C. Alonso-García, J. M. Ruiz, and W. Herrmann, “Computer simulation of shading effects in photovoltaic arrays,” Renewable Energy, vol. 31, no. 12, pp. 1986–1993, 2006.

[5] W. Xiao, W. G. Dunford, P. R. Palmer, and A. Capel, “Regulation of photovoltaic voltage,” IEEE Transactions on Industrial Electronics, vol. 54, no. 3, pp. 1365–1374, 2007.

[6] S. Rathore, A. Sharma, and R. Gupta, “Matlab/simulink-based modeling, simulation and performance evaluation of an improved maximum power point tracking algorithm for a

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10
***Article Received:** 26 August 2023 **Revised:** 20 October 2023 **Accepted:** 02 November 2023*

photovoltaic system," IET Renewable Power Generation, vol. 11, no. 7, pp. 897–906, 2016.

[7] J. Ahmed and Z. Salam, "An enhanced adaptive P&O MPPT for fast and efficient tracking under varying environmental conditions," IEEE Transactions on Sustainable Energy, vol. 9, no. 3, pp. 1487–1496, 2018.

[8] A. Urtasun, I. San Martín, P. Sanchis, and L. Marroyo, "Modeling of the electrical behavior of a photovoltaic panel by means of a multilayer perceptron," Renewable Energy, vol. 97, pp. 531–541, 2016.

[9] R. Ahmadi, H. Sayyaadi, and F. Soltani, "A new modeling and optimization approach for energy loss reduction in partially shaded PV arrays," Energy Conversion and Management, vol. 205, p. 112421, 2020.

[10] Z. Qu, G. Xiao, and W. G. Dunford, "Towards a PV panel diagnostic indicator via machine learning algorithms," Solar Energy, vol. 159, pp. 595–602, 2018.