People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research

University M'Hamed BOUGARA – Boumerdes

Institute of Electrical and Electronic Engineering

Department of Power and Control

Final Year Project Report Presented in Partial Fulfilment of
the Requirements for the Degree of the

MASTER

In **Control Engineering**

Option: **Control Engineering**

Title:

PV Power Forecasting Using Machine Learning With Hyperparameter Optimization

Presented by:

- **ZOUAOUI Samir**
- **BERIBECHE Abdessalem**

Supervisor:

Prof. Aissa KHELDOUN

Co-Supervisor:

Dr. Saida MAKHLOUFI

Registration Number:………./2023

Author’s Declaration of Originality

We hereby certify that we are the sole authors of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Authors:
ZOUAOUI Samir
BERIBECHE Abdessalem

11-07-2023

Abstract

The growing prominence of renewable energy, particularly photovoltaic (PV) power, necessitates accurate forecasting of PV power for both short and long-term horizons. Reliable forecasts are vital for effective decision-making and ensuring the stability of the electric grid. This thesis endeavors to analyze and compare a range of machine learning-based forecasting methods as alternatives to classical statistical time series forecasting techniques. Furthermore, the thesis presents a novel approach to hyperparameter tuning using metaheuristic algorithms with Differential Evolution and Particle Swarm Optimization. The models are evaluated based on their characteristics and performance, employing multiple metrics from existing literature. Additionally, a comparative study between different hyperparameter tuning algorithms is conducted. The investigation encompasses two distinct datasets and encompasses single-step and multi-step forecasting horizons.This thesis approach involves employing models to forecast the global irradiance, which is subsequently used to predict the output power of PV systems. By decoupling the prediction process into two stages, this method offers potential advantages in terms of accuracy and reliability.

Dedication

I would like to express my heartfelt dedication to
My parents,who have supported me unconditionally through both the good and bad times. I am grateful for the love, guidance, and support they have given me, as they have played a significant role in shaping the person I am today. I would also like to express my gratitude to my siblings for their unwavering support.

I extend my heartfelt appreciation to my friends, including Ikhlas Hachla, Omari Hamza, Salah Khouder, and Yacine Chabane, among others, who have helped me in countless ways. Though I may not be able to mention everyone individually, please know that I am sincerely grateful for your presence in my life. Your belief in me, even when I doubted myself, has been invaluable. I consider myself fortunate to have crossed paths with each of you. Thank you for everything.

Samir ZOUAOUI

I would like to dedicate my thesis to

the guiding light of my life, Allah. With unwavering faith, I acknowledge His blessings and mercy that empowered me throughout this academic journey.

To my beloved parents, whose unwavering support and sacrifices have been my constant motivation, I am forever grateful.

My dear brother and three sisters, your love and encouragement have been invaluable.

I extend my heartfelt gratitude to my incredible friends, whose unwavering belief in me pushed me beyond my limits. May Allah bless each one of you abundantly, for this achievement would not have been possible without your unwavering love and support.

Abdessalem BERIBECHE

Acknowledgments

In the name of Allah, the Most Beneficent and the Most Merciful, we thank Allah for all His blessing and strength that He gave us in completing this modest thesis.

We would like to express our deepest gratitude to our supervisor, Mr. Aissa Kheldoun, for his invaluable guidance, support, and expertise throughout the duration of this research project. His insightful feedback and unwavering commitment greatly contributed to the success of this study.

We would also like to extend our heartfelt appreciation to Mr. Hamza Belmadani for his assistance in providing us with valuable information on the metaheuristic search techniques. His expertise and willingness to share his knowledge played a significant role in enhancing our understanding of this aspect of the research.

Furthermore, we would like to express our sincere thanks to Madame Saida Makhloufi for generously providing us with the dataset from the CDER center and other valuable resources. Her contribution was instrumental in conducting a comprehensive analysis and achieving accurate results.

Lastly, we extend our gratitude to Miss Louiza Ait Mouloud, whose unwavering support and willingness to address our questions played a crucial role in guiding us towards the successful completion of this project. Her dedication and assistance were immensely appreciated.

We are indebted to all those mentioned above and would like to acknowledge their invaluable contributions, which have greatly enriched this research endeavor.

Samir ZOUAOUI, Abdessalem BERIBECHE

List of Abbreviations and Terms

AC	Alternating Current
AI	Artificial Intelligence
AR	AutoRegressive
ARIMA	Autoregressive Integrated Moving Average
CIGS	Copper Indium Gallium Diselenide
CdTe	Cadmium Telluride
CNN	Convolutional Neural Network
CDER	Centre de Développement d'Énergie Renouvelable
DC	Direct Current
DE	Differential Evolution
DKASC	Desert Knowledge Australia Solar Centre
DQN	Deep Q-Networks
ESN	Echo State Networks
GaAs	Gallium Arsenide
GRU	Gated Recurrent Units
HIT	Heterojunction with Intrinsic Thin Layer
IEEE	Institute of Electrical and Electronics Engineering
IQR	Interquartile Range
LSTM	Long Short-Term Memory
MA	Moving Average
MAE	Mean Absolute Error
MENA	Middle East and North Africa
ML	Machine Learning
NWP	Numerical Weather Prediction
NY	New York
PCA	Principal Component Analysis
PV	Photovoltaic
RBF	Radial Basis Function
RC	Reservoir Computing
RMSE	Root Mean Squared Error
RNNs	Recurrent Neural Networks
SARIMA	Seasonal Autoregressive Integrated Moving Average
SMA	Solar Multistring Architecture

SKTM	Shariket Kahraba wa Taket Moutadjadida
SSE	Sum of Squared Errors
SPSO	Standard Particle Swarm Optimization
SVM	Support Vector Machines
SVR	Support Vector Regression
TWh	Terawatt-hour
U.S.	United States
XGBoost	Extreme Gradient Boosting

Table of Contents

List of Figures

List of Tables

General Introduction

The increasing global emphasis on renewable energy sources has led to a significant rise in the utilization of photovoltaic (PV) power generation. In order to facilitate effective decision-making and ensure grid stability, accurate forecasting of PV power is crucial for both short and long-term horizons. The traditional statistical time series forecasting methods have been widely employed for this purpose. However, with the advent of machine learning techniques, alternative approaches have emerged, offering potential improvements in terms of accuracy and reliability.

One such alternative approach gaining prominence is the indirect forecasting method, which involves predicting the global irradiance as an intermediate step, and subsequently using this information to forecast the PV power output. This two-stage process decouples the prediction task, allowing for more focused and potentially more accurate results. By utilizing machine learning models for global irradiance prediction, the indirect forecasting method presents an opportunity to enhance the overall accuracy of PV power prediction.

In recent years, the field of time series forecasting has witnessed a significant shift towards the adoption of deep learning and machine learning methods. Notably, researchers such as Jimeng Shi, Mahek Jain, and Giri Narasimhan conducted a study that compared the performance of different recurrent neural network (RNN) architectures[1], with a specific focus on LSTM and Transformer models. Their research aimed to evaluate the efficacy of these architectures in capturing temporal dependencies and producing accurate forecasts for time series data.

Furthermore, investigations led by Qian Li, Zhou Wu, Rui Ling, Liang Feng, and Kai Liu explored the application of multi-reservoir echo state networks for solar irradiance[2] prediction. These studies leveraged the capabilities of these networks to capture complex dynamics and patterns inherent in solar irradiance data.

Moreover, the use of metaheuristic search algorithms for hyperparameter tuning in time series forecasting models has garnered attention. For instance, Lu Peng, Shan Liu, Rui Liu, and Lin Wang successfully employed Differential Evolution, a popular metaheuristic optimization algorithm, to tune the LSTM networks[3]. Their research demonstrated the efficacy of such approaches in enhancing the performance of deep learning models by effectively optimizing their hyperparameters.

In this study, we aim to analyze and compare various machine learning-based forecasting methods within the framework of the indirect forecasting approach. Specifically, we focus on the predictive performance and characteristics of these models.Furthermore, we explore the concept of hyperparameter tuning using meta-heuristic algorithms to optimize the performance of the machine learning models. This approach provides an opportunity to fine-tune the models' parameters for improved accuracy and robustness in PV power prediction.

To evaluate the models' performance, multiple metrics commonly employed in the literature are utilized. Moreover, a comparative study between different hyperparameter tuning algorithms is conducted to identify the most effective approach.

The thesis begins with Chapter One, which serves as an introduction to PV systems and explains why accurate PV power forecasting is crucial in the Energy industry.

In Chapter Two, we explore the field of time series analysis and introduce classical statistical methods. Additionally, we provide a brief overview of the machine learning models utilized in this study. This chapter also introduces the hyperparameter tuning algorithms employed, namely differential evolution and particle swarm optimization.

Chapter Three focuses on the datasets used in our research and describes how we performed preprocessing and feature engineering to ensure their suitability. We discuss the steps taken to prepare the data for analysis.

Finally, in Chapter Four, we present the results and findings of our study, followed by a comprehensive discussion of the outcomes. This chapter serves as the conclusion of the thesis, summarizing the key findings and their implications.

The outcomes of this research are expected to contribute to the advancement of accurate PV power forecasting and support informed decision-making in the evolving landscape of renewable energy. The findings will aid stakeholders, policymakers, and energy grid operators in improving the reliability and efficiency of PV power generation systems.

1. Review of Photovoltaic Systems

The following chapter delves into the world of photovoltaic systems, exploring their rich history, fundamental components, and diverse applications. By examining the origins and evolution of photovoltaics, we gain insights into the advancements that have shaped this renewable energy technology. We will then delve into the intricate workings of photovoltaic systems, shedding light on the photovoltaic effect and the crucial components that make up these systems. Additionally, we will explore the different types of photovoltaic systems and the various cell technologies that enable the conversion of sunlight into electrical energy. Finally, we will highlight the significance of PV output forecasting and its implications for optimizing the performance of these systems. Join us on this journey as we unravel the captivating world of photovoltaic systems and uncover their immense potential.

1.1 Brief history of Photovoltaic

The history of photovoltaic (PV) technology dates back to 1839 when French physicist Alexandre-Edmond Becquerel first demonstrated the photovoltaic effect. However, it wasn't until American inventor Charles Fritts created the world's first rooftop solar array in New York in 1883 that the potential of solar energy as an alternative to coal power was realized. The process of how light produces electricity wasn't understood until Albert Einstein explained the photoelectric effect in 1905, which laid the foundations for future developments in solar technology.[4]

The development of PV devices can be divided into three distinct phases. The first period, from 1839 to 1904, was a discovery phase without any real understanding of the science behind the operation of these first PV devices. In the second phase, from 1905 to 1950, a theoretical foundation for PV device operation and potential improvements was established. This period included Einstein's photon theory and the development of band theory for high purity single crystal semiconductors, which emphasized the importance of high purity single crystal semiconductors for high-efficiency solar cells.[5]

The practical silicon single-crystal PV device was developed in the third phase of PV device development between 1950 and 1959. This led to the creation of a variety of markets for PV, including powering remote terrestrial applications such as battery charging for navigational aids, signals, and telecommunications equipment. In addition, PV became a popular power source for consumer electronic devices in the 1980s, and efforts began to develop PV power systems for residential and commercial uses, both for stand-alone remote power as well as for utility-connected applications.[5]

Today, the industry's production of PV modules is growing at approximately 25 percent annually, and major programs in the U.S., Japan, and Europe are rapidly accelerating the implementation of PV systems on buildings and interconnection to utility networks. Overall, the history of PV technology is one of continuous innovation and improvement, with solar energy playing an increasingly important role in our global energy mix.[6]

1.2 Solar energy in the world

Photovoltaic (PV) technology has experienced significant growth and development in recent years, driven by declining costs, increasing efficiency, and a growing awareness of the need for sustainable energy sources. PV systems are becoming increasingly popular worldwide and are expected to continue playing a crucial role in the world's energy mix, contributing to the transition to a more sustainable and low-carbon energy system.

1.2.1 Globally

The growth of solar photovoltaic (PV) power generation achieved a new record in 2021, increasing by 179 TWh or 22% compared to the previous year. Despite this increase, solar PV still accounts for only 3.6% of global electricity generation and is the third largest renewable energy technology, behind hydropower and wind. Nonetheless, solar PV demonstrated remarkable resilience in 2021, overcoming Covid-19 disruptions, supply chain bottlenecks, and commodity price rises, and achieving another record annual increase in capacity of nearly 190 GW.[7]

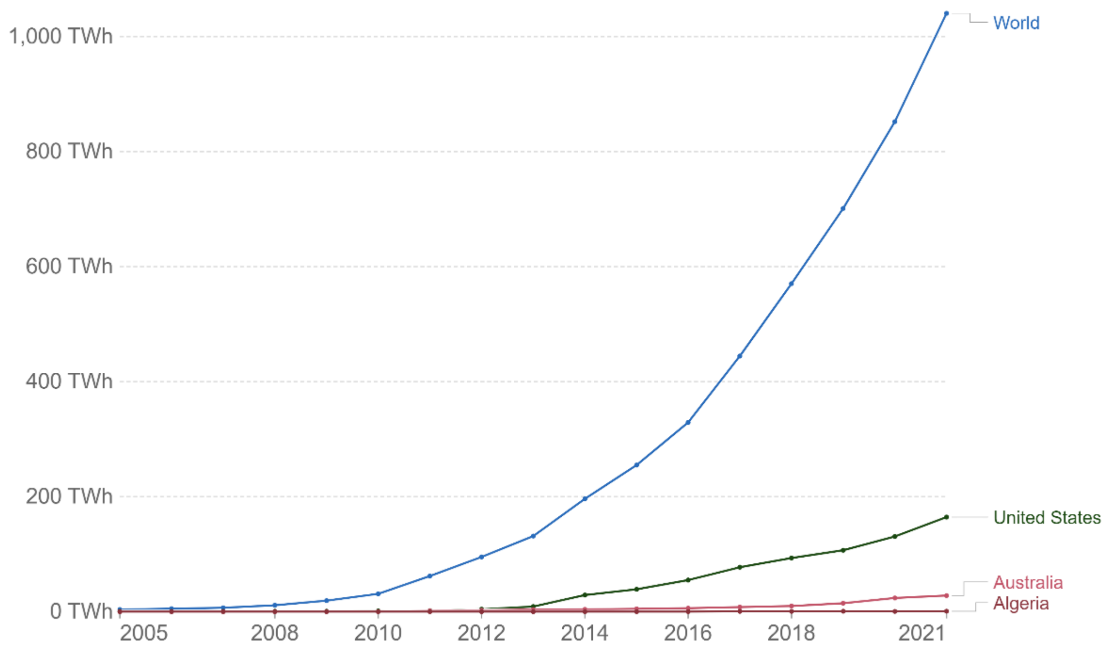

Figure 1. *Solar generation in the world[8].*

The shift towards solar energy is driven by its environmental benefits. Unlike traditional energy sources such as coal, oil, and natural gas, solar energy does not emit carbon dioxide, methane, or nitrous oxide, which are greenhouse gases responsible for polluting the atmosphere and contributing to climate change. As a result, solar energy is considered a sustainable and environmentally-friendly alternative to fossil fuels.[9]

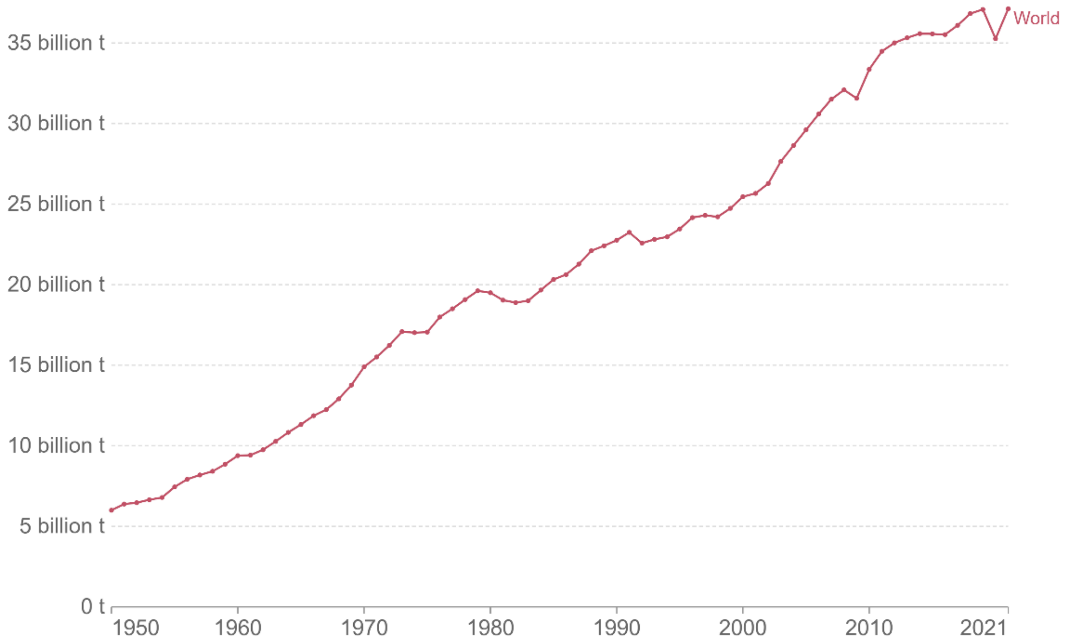

Figure 2. *Annual CO2 emissions[10].*

Solar energy has experienced a significant increase in capacity growth in recent years, with predictions of further acceleration of electricity generation growth in 2022. The growth of solar power has been particularly noteworthy, with the involvement of larger multinational corporations investing in renewable energy and undertaking significant solar projects. These developments have resulted in the setting of remarkable development targets. The progress of renewable energy, and in particular solar energy, has demonstrated an unexpected surge in growth, consistently outperforming expectations over the past decade.[11]

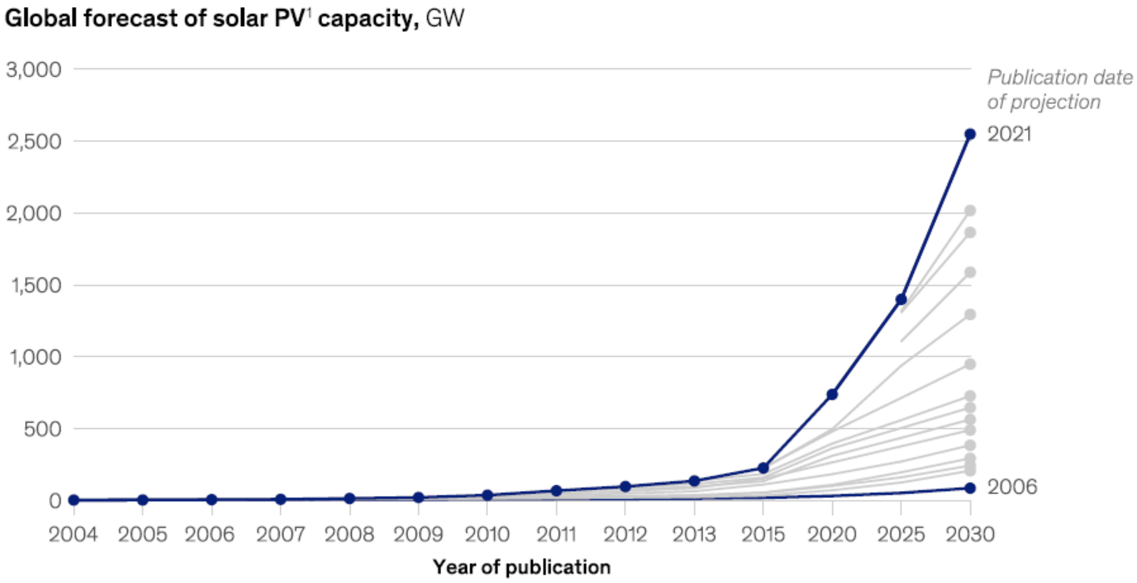

Figure 3. *Global prediction of solar PV capacity[11].*

1.2.2 In North Africa

The North African region is rich in solar energy, with high levels of solar radiation and clear skies throughout much of the year. The area experiences some of the greatest mean annual duration of bright sunshine, making it one of the highest level solar energy resources in the world. Countries in North Africa such as Algeria, Morocco, Tunisia, and Libya have huge potential to exploit this resource, with total solar annual irradiations ranging from 2,300 to 2,800 kWh/m2.[12]

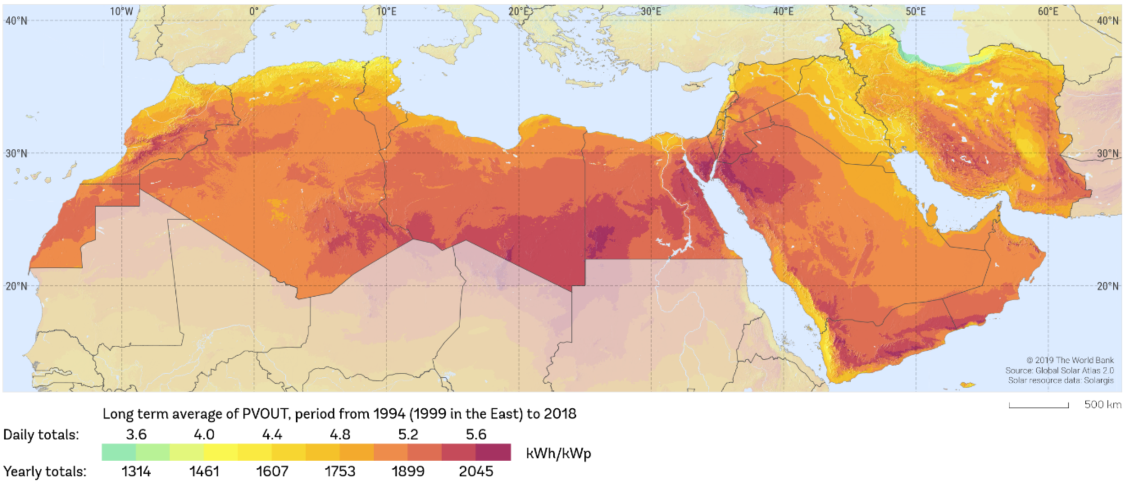
10°W
0°
10°E
20°E
30°E
40°E
50°E
60°E
40°N
30°N
20°N
© 2019 The World Bank
Source: Global Solar Atlas 2.0
Solar resource data: Solargis
500 km
Long term average of PVOUT, period from 1994 (1999 in the East) to 2018
Daily totals: 3.6 4.0 4.4 4.8 5.2 5.6
kWh/kWp
Yearly totals: 1314 1461 1607 1753 1899 2045

Figure 4. *Photovoltaic electricity potential in MENA region [13].*

However, the region has seen a 1.4 GW flood of new solar generation capacity added in 2019, the pace of new solar development has slowed significantly in recent years, with just 36 MW added in 2020. Nevertheless, North Africa's potential as a solar energy resource remains significant, and could play an important role in meeting global energy needs in the future.[14]

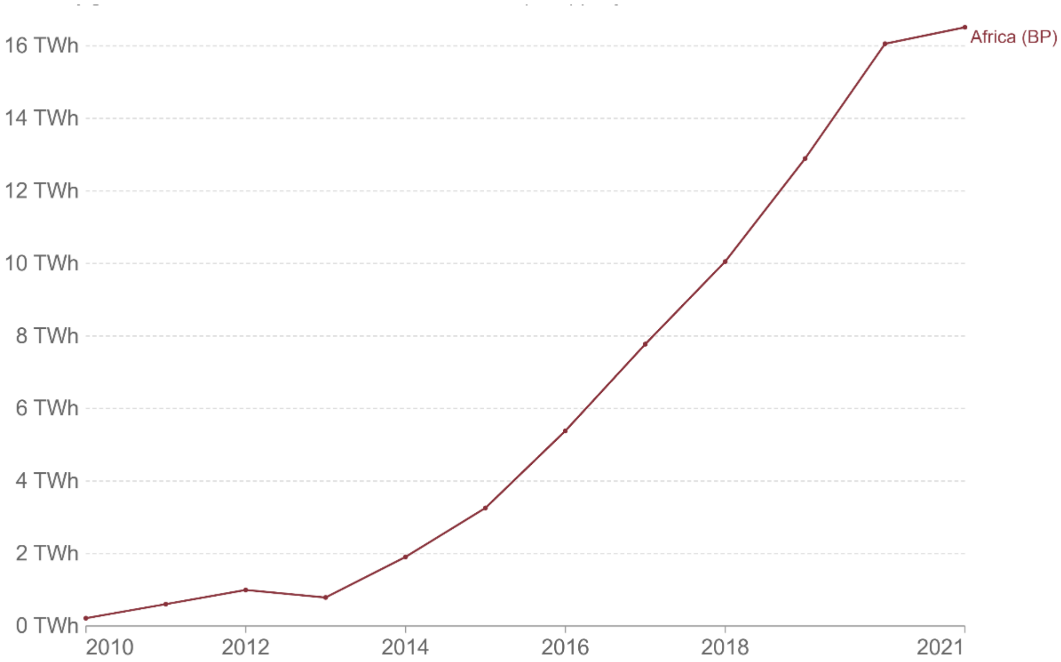

Figure 5. *Solar power generation in the Africa [15].*

1.2.3 In Algeria

Algeria, the second-most-populous country in North Africa and the largest nation in Africa by land area, holds significant potential for renewable energy [16]. Situated in the Sahara Desert, it benefits from ample sunlight year-round, making it an ideal location for harnessing solar energy. With an estimated capacity to produce 14TWh of energy annually from sunlight [17], Algeria has a valuable opportunity to shift towards renewable sources, reducing its reliance on traditional energy and enhancing macroeconomic stability [16].

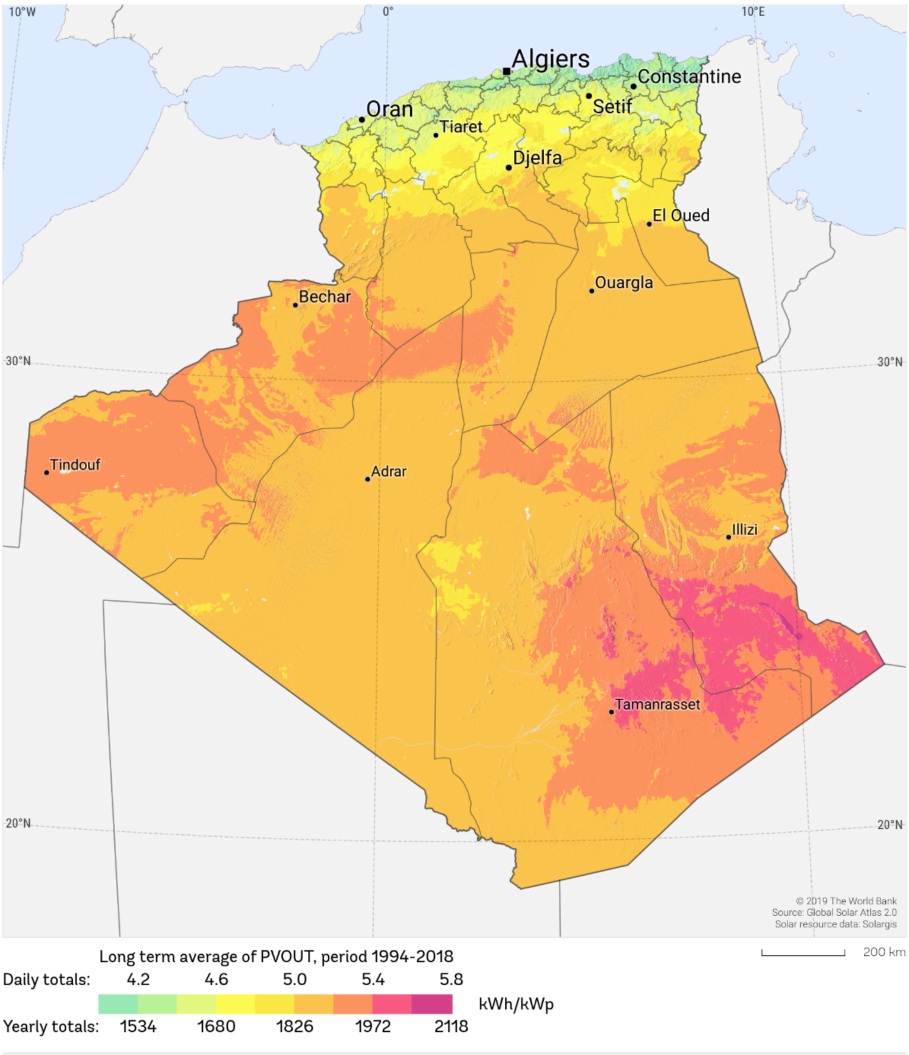
10°W
0°
10°E
30°N
20°N
Algiers
Constantine
Oran
Setif
Tiaret
Djelfa
El Oued
Ouargla
Bechar
Tindouf
Adrar
Illizi
Tamanrasset
© 2019 The World Bank
Source: Global Solar Atlas 2.0
Solar resource data: Solargis
200 km
Long term average of PVOUT, period 1994-2018
Daily totals: 4.2 4.6 5.0 5.4 5.8
kWh/kWp
Yearly totals: 1534 1680 1826 1972 2118

Figure 6. *Photovoltaic electricity potential in Algeria[18].*

To capitalize on its solar potential, Algeria has set an ambitious target of achieving 15,000 megawatts of solar energy by 2035, aiming to increase the current 3% share of solar-derived energy. This plan includes the installation of off-grid systems to benefit rural communities and promote sustainable development. Algeria's commitment extends beyond its domestic market, as it aspires to become a global competitor in the renewable energy marketplace, actively promoting renewable energy and energy efficiency programs[17].

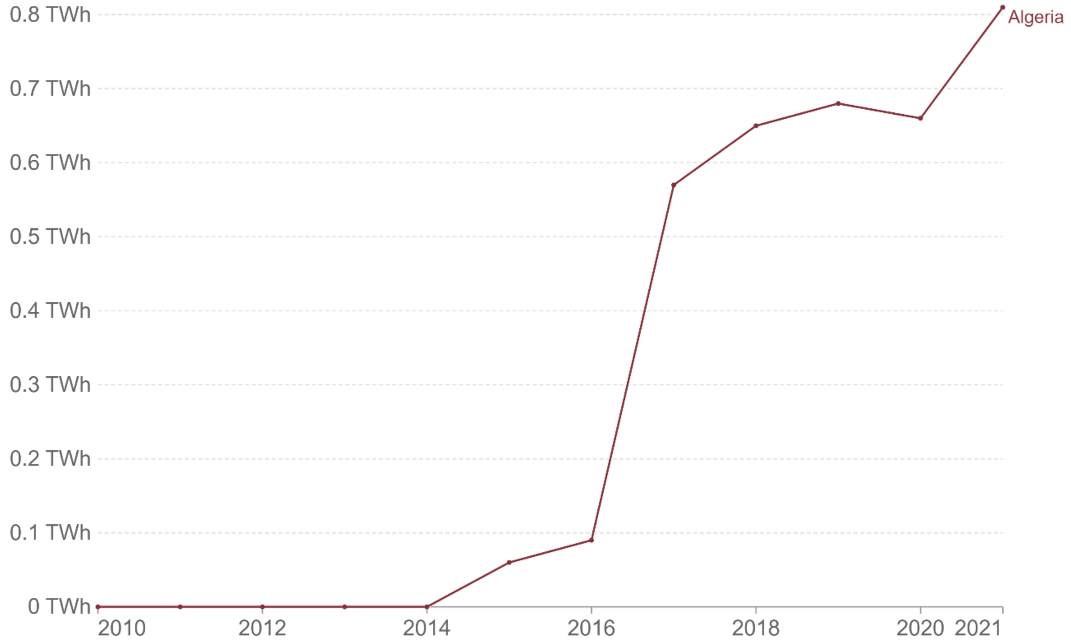

Figure 7. *Solar power generation in Algeria[19].*

Strategically located just 150 miles from Spain and Italy across the Mediterranean Sea, Algeria serves as a bridge between the MENA region, southern Europe, and sub-Saharan Africa. This geographic advantage positions Algeria as a potential energy supplier and a key connector between these regions, fostering opportunities for collaboration and energy trade[16].

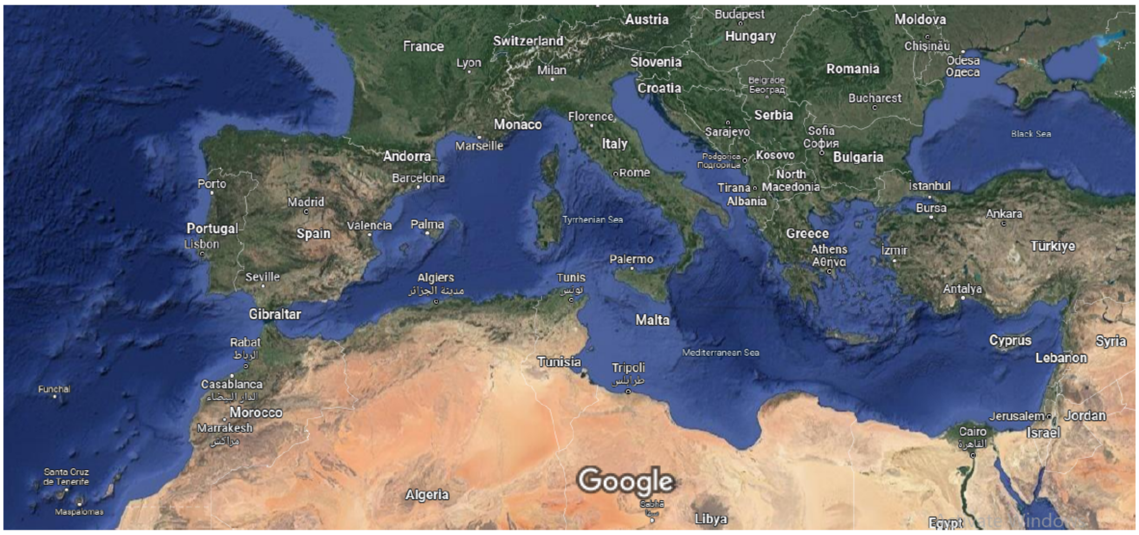

Figure 8. *Satellite View of Algeria's Geographical Location[20].*

The Algerian government recognizes the challenges associated with volatile oil and gas prices, increasing domestic electricity demand, and the urgency to address climate change. To address these challenges, Algeria has set ambitious targets, aiming to derive 27% of its electricity generation from renewable sources by 2030 and increase the share of renewable generation capacity to 37% by the same year[16].

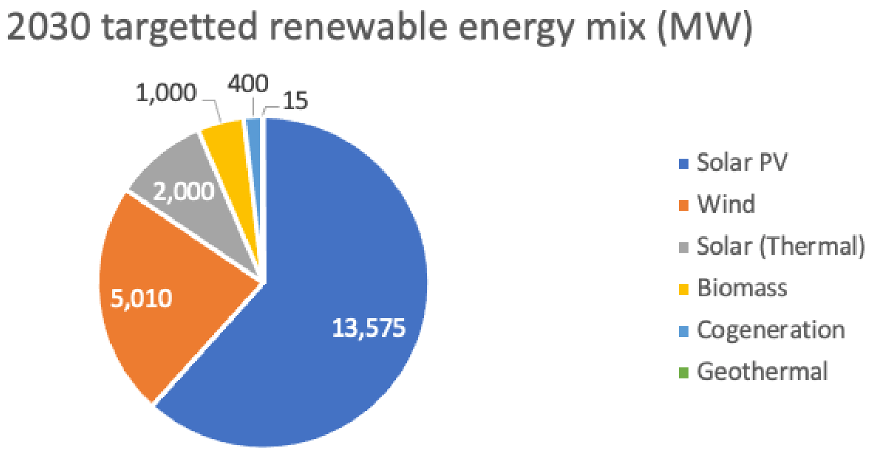

Figure 9. *2030 targeted renewable energy in Algeria[16].*

With its abundance of sunlight, ambitious renewable energy targets, and strategic geographical position, Algeria is poised to become a leading player in the development of solar energy. These factors create a highly favorable environment for the country to cultivate a thriving solar energy sector, promote energy trade, and contribute significantly to global efforts in combating climate change.

1.3 Photovoltaics

Photovoltaic plants are designed to convert sunlight directly into electrical energy. The fundamental component of these plants is the solar cell, which is made of silicon, a semiconductor material widely used in electronic devices such as diodes, transistors, and computer chips. The solar cell creates an electrical field in the crystal by introducing foreign atoms and creating a p-n junction. When sunlight strikes the solar cell, charge carriers are extracted from the crystal bindings and moved by the electrical field to the outer contacts, resulting in the generation of a voltage of approximately 0.5 V. The current released from the cell is dependent on the radiation and cell area and typically ranges between 0 and 10 A. To produce a usable voltage in the range of 20–50 V, multiple cells are connected in series to form a solar module. The efficiency of the photovoltaic system depends on various factors, including the quality of the solar cell, the angle and orientation of the solar panel, and the climatic conditions[21].

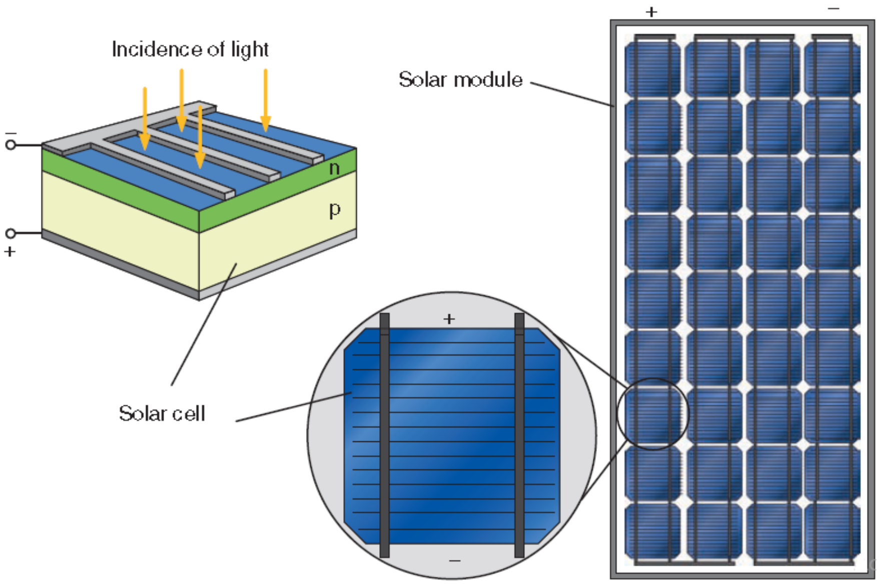

Figure 10. *The solar cell and solar module as basic components of photovoltaics[21].*

1.3.1 Photovoltaic system

A photovoltaic (PV) system composed of one or more solar panels, an inverter, and other electrical and mechanical hardware. The size of PV systems can range from small rooftop or portable systems to large utility-scale generation plants. Solar panels convert sunlight into direct current (DC) electricity, which is incompatible with the alternating current (AC) used by most electronic devices and the electrical utility grid. Therefore, an inverter is required to convert the DC electricity to AC electricity, which can then be used to power electronics locally or transmitted to the electrical grid.

In addition to solar panels, PV systems consist of several other essential components which are account for over half of the system cost and require the majority of maintenance such as:[22]

Solar Module:

A silicon crystalline module with multiple solar cells wired in series, mounted in an aluminum frame. Each cell produces 0.5 volts. The cell size determines the amperage produced. [23]

Solar Array:

A collection of PV modules wired in series to increase voltage, with each module's negative wire connected to the next module's positive wire. Multiple strings are connected to homerun leads and landed at an enclosure near the array to operate the loads in a residence[23].

Charge Controller:

Regulates charge flow from solar module to battery, preventing overcharging. Has amperage rating, may offer DC load connection and energy regulation based on battery charge level. Charges battery during day, powers load from battery at night. [23]

Battery:

Stores solar energy for use when the sun is not shining. Commonly used battery for residential PV is lead-acid. Deep-cycle battery is recommended for renewable energy systems, similar to a golf cart battery. Battery voltage varies from 2, 6, and 12 volts, while amp-hours can also vary. Dimensions of individual batteries can vary. [23]

Inverters:

electronic devices that convert DC energy into AC energy to power loads that operate on alternating current. In solar energy systems, string inverters are used to convert DC power from solar arrays into AC power, and are available in different sizes to match the AC load[23].

System Metering:

Tools are available to monitor solar energy systems, including battery meters for off-grid systems to measure energy input and output. Monitoring is important to ensure proper charging and discharging of batteries and system functionality. Web-based tools and apps allow for remote monitoring via cell phone or tablet[23]

Mounting System:

Refers to the apparatus used to secure solar arrays to rooftops or the ground. Typically made from steel or aluminum, they offer precise mechanical support for the panels. Some ground-mounted systems may include tracking systems that follow the sun's path in the sky, increasing energy output at a higher cost for equipment and maintenance.[22]

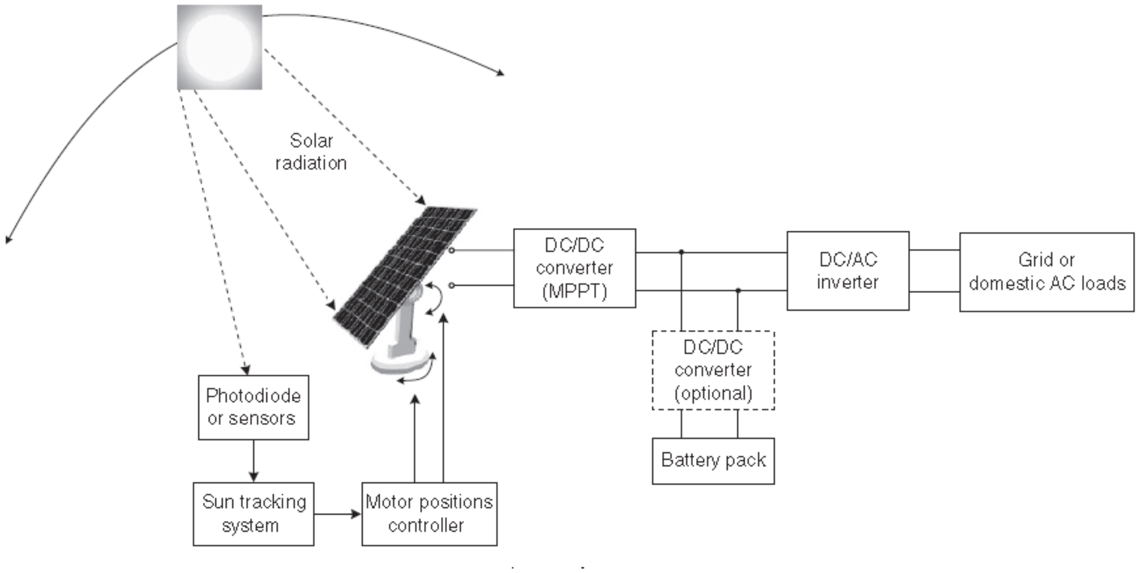

Figure 11. *Simplified Diagram: Understanding the Components of a PV System[24].*

Therefore, proper design and installation of a PV system are crucial to ensure efficient and reliable performance.

1.3.2 Types of PV systems

Photovoltaic power systems are commonly categorized based on their functional and operational specifications, the arrangement of their components, and their interconnection methods with other power sources and electrical loads. The primary classifications of these systems encompass grid-connected or utility-interactive systems, stand-alone systems and hybrid systems.

Grid-connected

PV systems typically do not require battery storage, but it is possible to add a battery to enhance the system.

(A) Grid-Connected PV Systems without Battery:

A grid-connected system utilizes a grid-tied inverter, making it suitable for residential solar installations. Net metering allows surplus energy to be redirected to the grid, resulting in lower energy bills. Solar panels convert solar radiation into DC energy, which is then converted to AC by the inverter for use by household devices. This type of system is cost-effective and offers design flexibility, but it does not provide outage protection.

(B) Grid-Connected PV Systems with Battery:

Integrating a battery into a grid PV system enhances energy independence. It reduces reliance on grid electricity and energy retailers, providing backup power when the solar system cannot generate sufficient energy.

A standalone

PV system operates independently without grid connection, making it suitable for rural areas. These systems require battery storage and can power applications like water pumps and ventilation fans. It's important to choose a reputable company for longer warranty coverage. However, designing a standalone system for household use requires meeting energy and battery charging needs. Some systems may include backup generators. Maintenance tasks include checking for terminal corrosion and monitoring battery electrolyte levels.

Hybrid

PV systems combine multiple power sources, such as wind, solar, or hydrocarbons, to enhance power availability. These systems often include a battery for improved efficiency. Benefits include independence from a single energy source and the ability to charge the battery when one source is insufficient. Hybrid PV systems are suitable for isolated areas with limited grid connections. Challenges include complex design and installation processes, as well as higher upfront.[25]

1.4 Photovoltaic Cell Types

PV cells use diverse methods and materials to convert solar energy into electrical power. Silicon, known for its semiconducting properties, is the primary material used for solar panel construction.

This report focuses on two specific PV cell technologies among the various available options.

Poly crystalline Silicon Cell

Poly-crystalline cells contain multiple small grains of crystals and are manufactured by casting a cube-shaped ingot using molten silicon. Another method, called edge-defined film-fed growth (EFG), involves extracting a thin ribbon of poly-crystalline silicon from molten silicon. While less efficient, poly-crystalline silicon PV cells are cost-effective and dominate the global PV market, representing around 70% of worldwide PV production in

2015.[26]

Hetero-junction with Intrinsic Thin layer (HIT)

HIT hybrid silicon technology is a highly efficient and adaptable solar cell technology. By combining amorphous and crystalline silicon, it achieves excellent conversion efficiencies and reliable performance in challenging conditions. HIT cells are known for their durability and resistance to microcracks, ensuring long-term reliability. Overall, HIT hybrid silicon technology represents a significant advancement in solar energy generation.[27]

1.5 PV output forecasting

As solar power becomes an increasingly integral part of our energy mix, accurate forecasting of its output has emerged as a critical necessity. Solar power generation poses unique challenges due to its reliance on meteorological variables, resulting in high volatility and operational complexities. To effectively plan, save costs, and ensure grid reliability, it is crucial to accurately forecast photovoltaic (PV) output. However, despite substantial investments in forecasts, there is a need to systematize scientific knowledge and analyze factors that impact forecast accuracy to provide reliable and objective insights.[28]

In contrast to conventional fossil fuel power generation, which can be dispatched predictably according to demand, renewable sources like solar power are heavily influenced by weather conditions. Consequently, the generation from these volatile sources fluctuates significantly. For example, PV output drops to zero during nighttime, affecting grid stability and electricity prices. Therefore, precise weather forecasts are essential for predicting the amount of electricity fed in from renewable sources and managing these consequences.[29]

The challenges faced in solar power forecasting go beyond the reliance on meteorological variables. Factors such as cloud detection, formation, dissipation, and atmospheric physics modeling further complicate forecast accuracy.[28]

Addressing load variability and compensating for imbalances caused by over- or under-generation by PV plants also pose challenges. Additionally, issues such as data availability, data quality, forecasting time horizons, and integrating forecasts with grid management systems need to be considered.[30]

To overcome these challenges, significant advancements have been made in solar forecasting techniques. Research and development efforts have primarily focused on forecasting power generated by variable energy resources such as wind and solar. Notably, refinements in Numerical Weather Prediction (NWP) models, the utilization of infrared satellite imagery, and the adoption of advanced data processing methods like deep machine learning have contributed to improved solar forecasting accuracy. Furthermore, the shift from deterministic to probabilistic forecasts has further enhanced the precision of predictions.[30]

Operational utilization of solar forecasts is now prevalent, especially in regions with high PV penetration. Accurate forecasting of solar irradiance, accounting for its variability, is vital for the efficient integration of solar power into the grid. This requires accurate spatial and temporal forecasting of solar irradiance for horizons ranging from 1-6 hours to day-ahead. Network operators need advanced modeling capabilities to predict the output of numerous PV plants accurately. Moreover, the estimation of forecast uncertainty is of utmost importance to ensure reliable grid operation.[30]

In conclusion, continuous research and improvement in solar forecasting techniques are indispensable for enhancing the reliability and effectiveness of integrating solar power into the grid. Collaboration between researchers, industry professionals, and policymakers is essential to address the challenges, explore practical solutions, and ensure a sustainable and efficient future powered by solar energy.

1.6 Conclusion

In this chapter, we explored the fundamentals of photovoltaic (PV) systems, including the energy potential of PV power globally, in North Africa, and specifically in Algeria. We discussed the science behind solar cells, the components of a basic PV system, and the different types of PV technologies. We also touched upon the challenges of grid integration and highlighted the importance of PV power forecasting for efficient and stable operation. This foundation sets the stage for further research on machine learning-based forecasting techniques to enhance the utilization of solar energy.

2. Time Series Analysis and Machine Learning Models for PV Power Forecasting.

The problem of PV power forecasting, while of significant importance in the renewable energy domain, is not a unique challenge in isolation. It falls within a broader class of problems known as time series forecasting. As PV power generation is inherently a time-dependent process, it is crucial to explore the characteristics and methodologies of time series analysis to effectively tackle this forecasting problem. This chapter aims to provide a comprehensive academic exploration of time series analysis techniques and their application in PV power forecasting. Additionally, it delves into the transition from traditional statistical techniques to machine learning and deep learning approaches, investigating their suitability and performance in this domain.

To commence our journey, it is vital to establish a fundamental understanding of time series. In technical terms, a time series refers to a sequence of data points collected at regular intervals over time. Understanding the properties and behaviors of time series data is essential for accurate forecasting. We will delve into the nature of time series data, examining its temporal dependencies, trends, and seasonality. Furthermore, we will investigate the statistical methods used for forecasting, such as ARIMA and Exponential smoothing.

However, as technology advances and computational capabilities increase, machine learning models have emerged as powerful tools for time series forecasting. By leveraging the capacity of these models to capture complex relationships and patterns, researchers have achieved notable improvements in forecasting accuracy. In this chapter, we will shift our focus towards the application of machine learning models to PV power forecasting.

We will embark on a comprehensive exploration of commonly employed machine learning models for PV power forecasting. These models include linear regression, support vector regression (SVR), XGBoost, decision trees, random forests, Long Short-Term Memory (LSTM) networks, Convolutional Neural Network Long Short-Term Memory (CNN-LSTM), bidirectional LSTM, and reservoir computing. By examining the theoretical underpinnings and practical implications of these models, we aim to provide the reader with valuable insights into their suitability and performance in the context of PV power forecasting.

In addition to exploring the individual machine learning models, we will also investigate the process of mapping the time series forecasting problem to a supervised learning framework. This process involves transforming the time series data into a structured format suitable for training machine learning models. In summary, this chapter serves as a comprehensive guide to the fundamental concepts of time series analysis and machine learning models for PV power forecasting. Through an exploration of various algorithms and their applicability to the PV domain, we strive to contribute to the advancement of accurate and reliable PV power predictions, fostering the integration of renewable energy sources into the electrical grid and supporting the transition towards a sustainable energy future.

2.1 Introduction to Time series Analysis

2.1.1 Definition and Basics

In the field of mathematics, a time series refers to a collection of data points that are arranged or represented in chronological order. Typically, a time series consists of a sequence of data points captured at regular intervals over time, thereby forming a set of discrete-time data[1].

The time series can be formally defined as the set

$$U = \{x_1, x_2, x_3, \ldots, x_t\}, \tag{2.1}$$

where the index $m = 1, 2, 3, \ldots, T$ represents the time index at equal sampling intervals.

In the field of time series analysis, sequences can be classified into two fundamental categories: deterministic and nondeterministic. A deterministic time series refers to a sequence in which its evolution can be fully described using mathematical expressions, enabling a precise understanding of its behavior over time. On the other hand, a nondeterministic time series is characterized by the absence of an explicit analytic expression to describe its behavior. It exhibits a random aspect that impedes the ability to provide a comprehensive description of its evolution. Hence, A nondeterministic time series is a stochastic process indexed by integers.

2.1.2 Characteristics of Time series

Prior to constructing a model for a time series, it is essential to thoroughly examine its characteristics. Time series data exhibits various types, which can be classified based on their components and evolving patterns, such as seasonality, stationary, and non-stationarity. Understanding these characteristics is crucial in order to make informed modeling decisions and apply appropriate analytical techniques.

Seasonality

Seasonality denotes the periodic fluctuations observed in a time series. For instance, the daily variation in global irradiance demonstrates this pattern, with higher levels during the day and lower levels at night. Such recurring patterns can be visually depicted in figure 12.

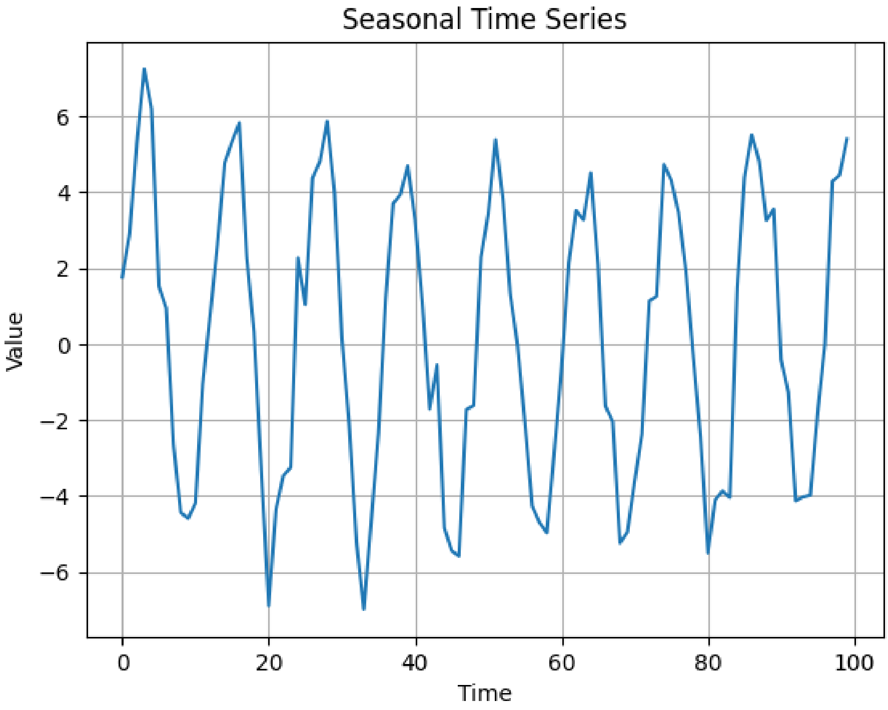

Figure 12. *Seasonal Time series.*

Stationarity

Stationarity is a significant attribute associated with time series data. A time series is considered stationary when its statistical characteristics remain constant over time. This implies that the series maintains a consistent mean and variance, while the covariance remains independent of time[31].

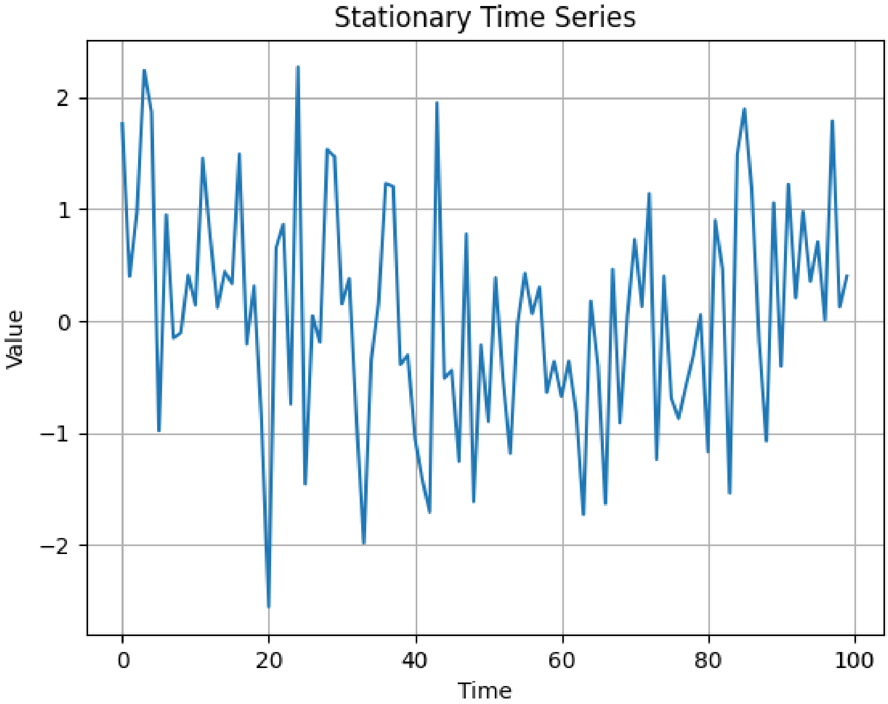

Figure 13. *An example of a stationary Time series.*

A time series that does not meet the conditions of stationarity is referred to as a non-stationary time series.

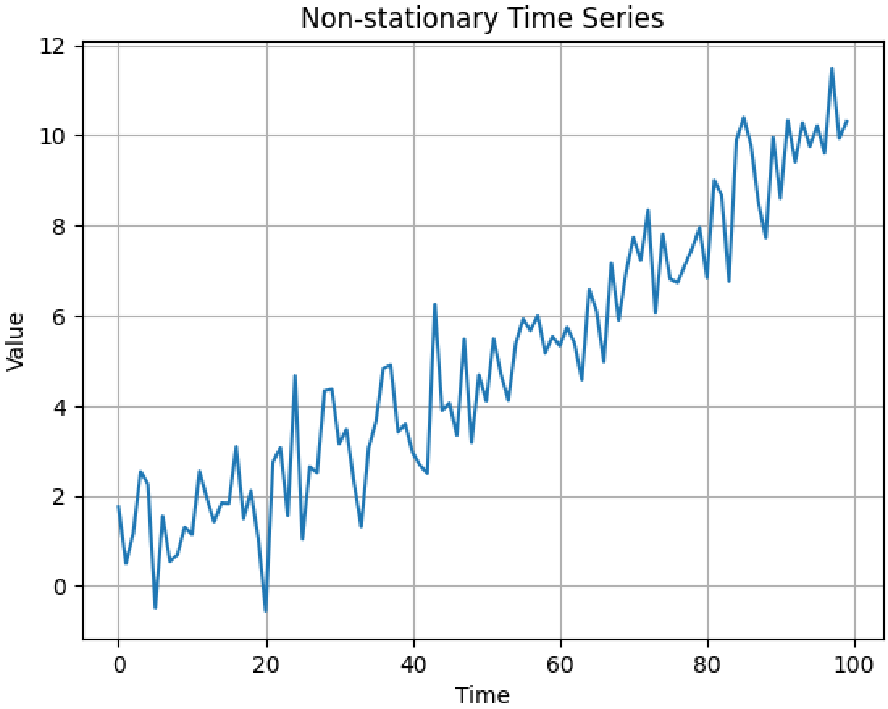

Figure 14. *An example of non-stationary Time series.*

2.1.3 Time series Forecasting

Time series forecasting involves analyzing historical data in a sequential order to identify patterns, trends, and seasonality. By leveraging this information, accurate predictions of future values or patterns can be made.

Time series forecasting can be classified into different types based on the nature of the forecasted values and the forecasting approach:

Univariate Time Series Forecasting

Univariate time series forecasting focuses on predicting the future values of a single variable over time. It involves analyzing historical data for that specific variable and using it to make predictions.

Multivariate Time Series Forecasting

Multivariate time series forecasting involves predicting the future values of multiple variables simultaneously. It considers the dependencies and relationships between these variables to generate more comprehensive and accurate forecasts.

Time series forecasting techniques can further be categorized based on the forecasting horizon:

Single Step Forecasting

Single step forecasting predicts the value of a time series variable at the next time step based on past observations. It focuses on forecasting one time step ahead.

Multi-Step Forecasting

Multi-step forecasting aims to predict the future values of a time series variable over multiple time steps. It provides forecasts beyond the immediate next step, allowing for insights into longer-term trends and patterns.

2.1.4 Modeling Time series

The modeling of nondeterministic time series presents a challenging problem, as their behavior cannot be fully determined in advance. To address this challenge, mathematicians have developed various statistical methods aimed at modeling and approximating time series in order to make predictions. Some of these notable statistical methods include:

Moving Averages

Moving Average (MA) is a statistical method commonly used in time series analysis to identify and capture underlying patterns or trends in data. It involves calculating the average of a fixed number of preceding observations to generate a smoothed estimate of the current data point.

Mathematically, the moving average of a time series can be defined as follows:

Given a time series $\{X_1, X_2, X_3, ..., X_t\}$, where X_t represents the data point at time t, the moving average of order q at time t (MA(q)) is calculated as:

$$MA(q)_t = \frac{X_t + X_{t-1} + X_{t-2} + ... + X_{t-q}}{q} \tag{2.2}$$

In this formulation, q represents the order or number of preceding observations included in the moving average calculation. The moving average provides a smoothed representation of the time series by averaging out the short-term fluctuations and highlighting the long-term trends or patterns present in the data.

By adjusting the order (q) of the moving average, analysts can control the level of smoothing applied to the time series. Higher values of q result in a greater level of smoothing, while lower values preserve more of the short-term variations in the data.

A moving average is commonly used with time series data to smooth out short-term fluctuations and highlight longer-term trends or cycles.

From a mathematical standpoint, a moving average can be understood as a form of convolution, making it akin to a low-pass filter employed in signal processing. Although it is commonly associated with time series data, when applied to non-time series data, a moving average serves as a filter for attenuating higher frequency components. While the ordering of the data typically implies some form of temporal relationship, the application of a moving average does not necessitate a specific connection to time[32].

Exponential Smoothing

Exponential smoothing employs a comparable rationale to moving average, but with a distinct approach of assigning diminishing weights to individual observations. In simpler terms, as we move away from the present, the significance of observations gradually diminishes[31].

Mathematically, the exponential smoothing equation can be expressed as follows:

$$S_0 = X_0$$

$$S_t = \alpha \cdot X_t + (1 - \alpha) \cdot S_{t-1} \tag{2.3}$$

Where: S_t is the smoothed value at time t, X_t is the observed value at time t, S_{t-1} is the smoothed value at time $t - 1$, and α is the smoothing parameter ($0 \leq \alpha \leq 1$) that determines the weight given to the current observation. The smoothed value can be used as the forecasted value for the next time period[31].

AutoRegressive Integrated Moving Avergae (ARIMA)

ARIMA (AutoRegressive Integrated Moving Average) is a popular time series analysis model used to forecast future values based on historical data. It combines the concepts of autoregressive (AR), differencing (I), and moving average (MA) models to capture the underlying patterns and dynamics of a time series. ARIMA models are widely utilized in various domains, such as economics, finance, and epidemiology, for their ability to handle both seasonal and non-seasonal data.

The mathematical equations for ARIMA can be expressed as follows:

1. AutoRegressive (AR) Component: AR(p) represents the autoregressive component, which captures the linear relationship between the current observation and a certain number of lagged observations. It can be defined as:

$$X_t = c + \phi_1 X_{t-1} + \phi_2 X_{t-2} + ... + \phi_p X_{t-p} + \varepsilon_t \tag{2.4}$$

where X_t is the value of the time series at time t, c is a constant, ϕ_i represents the coefficients of the lagged observations, p denotes the order of the autoregressive component, and ε_t is the error term at time t.

2. Integrated (I) Component: The integrated component of ARIMA focuses on differencing the time series to achieve stationarity. It can be represented as:

$$Y_t = X_t - X_{t-1} \tag{2.5}$$

where Y_t denotes the differenced series.

3. Moving Average (MA) Component: The moving average component represents the weighted sum of past error terms. It can be expressed as:

$$X_t = \mu + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + ... + \theta_q \varepsilon_{t-q} + \varepsilon_t \tag{2.6}$$

where μ is the mean of the time series, θ_i are the coefficients of the lagged error terms, q indicates the order of the moving average component, and ε_t represents the error term at time t.

ARIMA models can effectively handle both seasonal and non-seasonal time series data by incorporating additional seasonal terms and differencing. By identifying and incorporating seasonal patterns, ARIMA models can capture and forecast seasonal variations in the data. Additionally, the integration component allows for differencing to achieve stationarity, which is crucial for accurate modeling and forecasting.

2.1.5 Limitations of Traditional Statistical Models in Time Series Forecasting and the Rise of Machine Learning Approaches

Statistical models, including those mentioned, are based on the assumption of linearity, assuming a linear relationship between past observations and future steps. However, this assumption does not always hold true, as time series data often exhibit non-linear patterns. Additionally, statistical models are known to be sensitive to outliers and struggle to handle seasonal data. Although models like SARIMA have been developed to address seasonality, they still rely on the aforementioned assumptions and suffer from the same limitations. Furthermore, these models were primarily designed for univariate time series forecasting, making it challenging to generalize them to multivariate cases. Consequently, the scientific community has increasingly turned to the utilization of machine learning approaches, which offer more flexibility and effective solutions to these problems. Machine learning methods provide greater capability to capture non-linear patterns, robustness against outliers, and the ability to handle seasonality and multivariate time series data more seamlessly. Thus, the adoption of machine learning approaches has gained prominence due to their ability to overcome the limitations of traditional statistical models in time series forecasting.

2.2 Machine Learning for Time Series Forecasting

2.2.1 Definition of Machine Learning

Machine learning, as defined by Mitchell (1997), can be precisely described as the process of enhancing a measure of performance P for a given task T through the utilization of training experience E [33].

Machine learning is a branch of artificial intelligence (AI) and computer science that focuses on developing models using data. It leverages algorithms and data to simulate human learning processes and improve accuracy over time. Within this context, learning involves transforming experiences into knowledge or expertise. Learning algorithms take training data, which represents accumulated experiences, as input and typically produce a computer program capable of executing specific tasks as output.

Numerous applications make use of machine learning techniques, including:

1. Automatic classification of products by analyzing images on a production line.
2. Tumor detection in brain scans.
3. Automated identification of offensive comments in social media platforms.
4. Automatic summarization of lengthy documents.
5. Development of chatbots or personal assistants.
6. Recommendation systems for personalized suggestions.
7. Detection of credit card fraud.
8. PV energy forecasting to facilitate the transition towards a greener power grid.

By harnessing the power of data and algorithms, machine learning enables improved efficiency and accuracy in various domains, contributing to advancements in AI and computer science.

2.2.2 Machine Learning Paradigms

Machine learning encompasses three main paradigms: supervised learning, unsupervised learning, and reinforcement learning.

Supervised Learning

Supervised learning involves training models using labeled data, where input data is paired with corresponding output labels. The goal is to predict or classify new, unseen data based on the patterns and relationships learned from the labeled examples. Common models used in supervised learning include linear regression, support vector machines (SVM), decision trees, and random forests.

Unsupervised Learning

Unsupervised learning deals with unlabeled data and focuses on discovering patterns, structures, or relationships within the data itself. Clustering and dimensionality reduction are common tasks in unsupervised learning. Models such as k-means clustering, hierarchical clustering, and principal component analysis (PCA) are commonly used in this type of learning.

Reinforcement Learning

Reinforcement learning involves an agent learning through interaction with an environment to maximize rewards or achieve specific goals. The agent takes actions, receives feedback in the form of rewards or penalties, and learns from the consequences of its actions. Popular algorithms used in reinforcement learning include Q-learning and deep Q-networks (DQN).

These different paradigms of machine learning provide a range of approaches to solve diverse problems, including prediction, classification, clustering, and decision-making. The choice of which paradigm to use depends on the nature of the problem, available data, and desired outcome.

2.2.3 Regression

Regression analysis attempts to estimate the relationship among variables using a variety of statistical tools. Specifically, one can consider the general relationship between independent variables X, dependent variables Y, and some unknown parameters β:

$$Y = f(X, \beta) \tag{2.7}$$

where the regression function $f(\cdot)$ is typically prescribed, and the parameters β are found by optimizing the goodness-of-fit of this function to data [34].

Regression is a fundamental problem in machine learning, falling under the umbrella of supervised learning.

2.2.4 Time Series Forecasting as Regression

The PV power forecasting problem, along with various other time series forecasting tasks, can be formulated as regression problems within the framework of supervised learning. Although time series data has a temporal nature, it can be transformed into a suitable format for regression analysis. This transformation is achieved by designing appropriate features that capture the temporal dependencies and other relevant characteristics of the time series. By mapping the time series data into a supervised learning setting, where past observations serve as input features and future observations act as target variables, regression models can be trained to make accurate predictions based on historical patterns.

To transform a time series into a regression supervised learning problem, a common approach is to use a technique called matrix embedding. The matrix embedding represents the temporal dependencies between past observations and future observations by constructing a matrix of input features and target variables [35].

Let's consider a time series with n observations, denoted as $X = [x_1, x_2, \ldots, x_n]$. We can create a matrix embedding by selecting a suitable time lag, denoted as m, which determines the number of past observations considered as input features for each target variable. The matrix embedding can be represented as follows:

$$\begin{bmatrix} x_1 & x_2 & \ldots & x_m & x_{m+1} \\ x_2 & x_3 & \ldots & x_{m+1} & x_{m+2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{n-m+1} & x_{n-m+2} & \ldots & x_{n-1} & x_n \end{bmatrix}$$

In this matrix, each row represents a set of input features, consisting of m past observations, and the corresponding target variable is represented by the column on the right.

By adjusting the time lag m, we can control the extent of temporal dependencies captured by the regression model. A smaller time lag captures more immediate dependencies, while a larger time lag incorporates longer-term dependencies.

By utilizing the time embedding matrix, regression models can learn the underlying patterns and relationships in the time series data. The models can then use this knowledge to generate accurate predictions for future time steps.

Overall, transforming time series into a regression supervised learning problem allows us to leverage the capabilities of regression models to make accurate predictions based on historical patterns and temporal dependencies captured in the time embedding matrix.

2.3 Machine Learning Models for PV Power Forecasting

In this section, we focus on the machine learning models utilized for PV power forecasting. We explore both linear and non-linear models, examining their performance in relation to the complexity of the models. The models are organized in a hierarchy, ranging from the least complex to the most complicated. Linear models, such as linear regression and support vector regression, are initially considered due to their simplicity and interpretability. Subsequently, we delve into non-linear models, including decision trees, random forests, and neural networks, which are known for their ability to capture complex patterns and relationships within the data. By exploring models of varying complexities, we aim to identify the most suitable approach for achieving accurate and reliable PV power forecasting.

2.3.1 Linear Regression

Linear regression is a statistical modeling technique that aims to establish a linear relationship between a dependent variable and one or more independent variables. It assumes that this relationship can be represented by a straight line. The goal of linear regression is to find the best-fitting line that minimizes the distance between the observed data points and the predicted values.

In matrix notation, the linear regression equation can be defined as:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \tag{2.8}$$

where:

- $\mathbf{y}$ is the vector of dependent variables,
- $\mathbf{X}$ is the matrix of independent variables,
- $\boldsymbol{\beta}$ is the vector of coefficients (slopes) associated with the independent variables,
- $\boldsymbol{\varepsilon}$ represents the error term, which accounts for the deviation between the observed and predicted values.

The linear regression equation can be further expanded as:

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_{11} & x_{12} & \dots & x_{1p} \\ 1 & x_{21} & x_{22} & \dots & x_{2p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & \dots & x_{np} \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}$$

where:

- n is the number of observations,
- p is the number of independent variables,
- y_i represents the observed value of the dependent variable for the ith observation,
- x_{ij} represents the jth independent variable for the ith observation,
- β_j represents the coefficient associated with the jth independent variable,
- ε_i represents the error term for the ith observation.

The goal of linear regression is to estimate the values of the coefficients $\boldsymbol{\beta}$ that minimize the sum of squared errors (SSE) between the observed and predicted values. This is typically done using the least squares method.

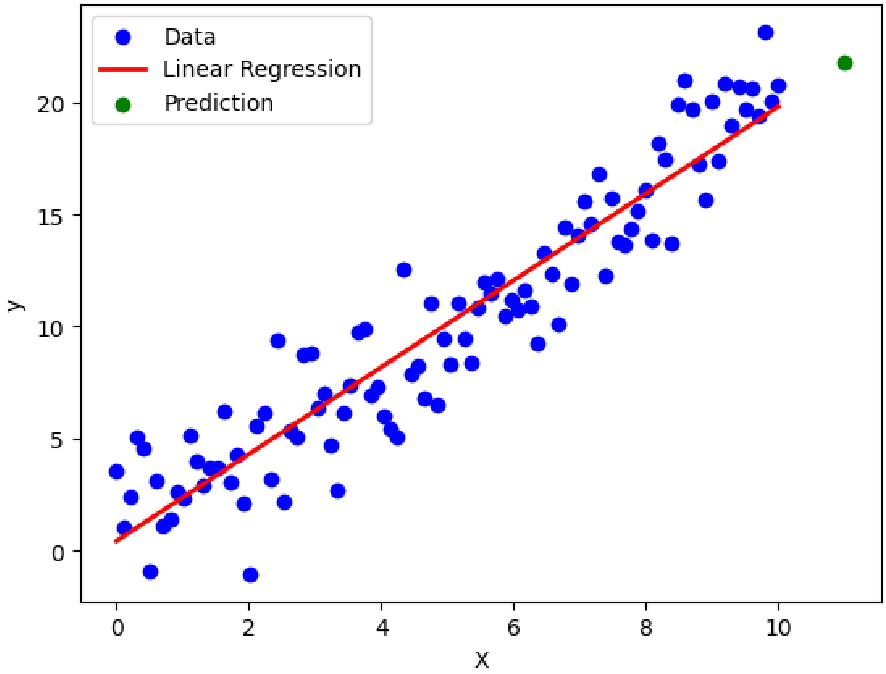

Figure 15. *Linear Regression Model.*

2.3.2 Support Vector Machine

Support Vector Regression (SVR) is a machine learning algorithm used for regression tasks. It is based on the principles of Support Vector Machines (SVM) and aims to find a hyperplane that best fits the given data while minimizing the prediction error.

In SVR, the objective is to find a function $f(\mathbf{x})$ that predicts the continuous target variable y given the input vector $\mathbf{x}$. The SVR model is defined by the equation:

$$y = f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b \tag{2.9}$$

where:

- y is the target variable,
- $\mathbf{x}$ is the input vector,
- $\mathbf{w}$ represents the weight vector,
- b is the bias term.

The goal of SVR is to find the optimal values for $\mathbf{w}$ and b that minimize the error between the predicted values and the actual values.

SVR introduces the concept of a margin, which determines the tolerance for errors. It seeks to find a hyperplane that maintains a maximum margin while still allowing a certain degree of error. The margin is defined by the support vectors, which are the data points closest to the hyperplane.

To handle non-linear relationships between the input features and the target variable, SVR utilizes a kernel function. The kernel function maps the input features into a higher-dimensional space, where a linear hyperplane can be constructed to capture complex patterns in the data.

The choice of the kernel function, as well as the hyperparameters such as the regularization parameter and the kernel coefficient, significantly impact the performance of SVR. Commonly used kernel functions include the linear, polynomial, radial basis function (RBF), and sigmoid kernels.

The training of SVR involves solving a constrained optimization problem, where the goal is to minimize the prediction error while satisfying the margin constraints. Various optimization techniques, such as quadratic programming, are employed to find the optimal solution.

SVR is a powerful algorithm for regression tasks, particularly when dealing with non-linear relationships. It can effectively handle datasets with high-dimensional features and is robust to outliers.

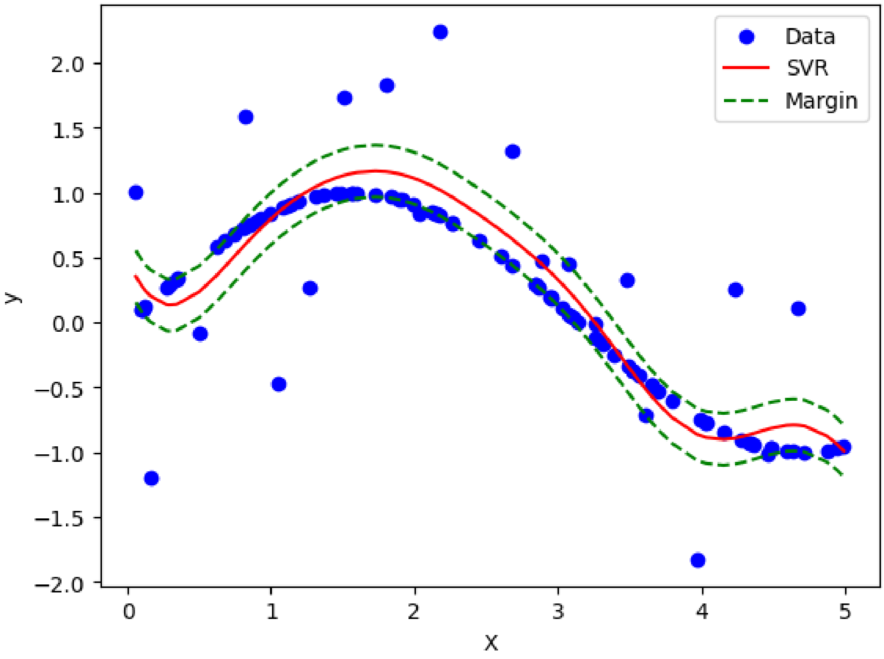

Figure 16. *Support Vector Regression with RBF kernel.*

2.3.3 Decision Trees Regression

Decision tree regressors are non-parametric supervised learning models used for predicting continuous target variables. They create a hierarchical structure of nodes, where internal nodes represent feature-based decisions and leaf nodes correspond to predicted values.

Data is recursively split based on optimal feature separation, using criteria like mean squared error or variance reduction. The resulting tree-like structure enables predictions for new data.

Advantages of decision tree regressors include interpretability, capturing non-linear relationships, handling missing values without imputation, and robustness to outliers.

However, drawbacks include overfitting with deep and complex trees, producing piecewise constant predictions, and sensitivity to small changes in training data.

Despite limitations, decision tree regressors are widely used. Techniques like pruning, ensemble methods (e.g., Random Forests), and regularization can improve performance and mitigate overfitting[36].

2.3.4 Random Forests

Random forests are highly regarded and effective machine learning models used for classification and regression tasks. They build multiple decision trees on randomly selected subsets of the training data, overcoming limitations of individual trees and improving accuracy and robustness.

Advantages of random forests include:

- High accuracy: By leveraging collective knowledge, random forests yield accurate predictions.
- Robustness to overfitting: Randomization in feature selection and data sampling reduces overfitting, promoting better generalization.
- Handling high-dimensional data: Random forests effectively handle datasets with numerous features, eliminating the need for feature selection.
- Variable importance: They assess feature importance, providing insights into relevance.

However, random forests have limitations:

- Interpretability: The ensemble model is less interpretable than individual trees.
- Computational complexity: With numerous trees and features, random forests can be computationally demanding.
- Model size: They have larger model sizes compared to single decision trees.

Despite limitations, random forests are widely used, performing exceptionally well in complex and high-dimensional data. Parameter tuning and optimizing the ensemble size enhance their effectiveness. [37]

2.3.5 XGboost

XGBoost, or Extreme Gradient Boosting, is an efficient implementation of the gradient boosting ensemble algorithm that excels in both classification and regression tasks. It has gained significant popularity in the data science community and has become a favorite among practitioners and competition winners, such as those on Kaggle[38].

Noteworthy for its speed and efficiency, XGBoost delivers exceptional performance across a wide range of predictive modeling problems. By combining decision trees and gradient boosting, XGBoost continually improves the model by iteratively adding trees that correct the errors of the existing ones. This iterative process minimizes a specific loss function, enabling XGBoost to achieve impressive accuracy and robustness[38].

While primarily designed for tabular datasets, XGBoost can also be utilized for time series forecasting, making it a versatile tool in predictive analytics. However, it is important to note that for time series forecasting, the time series data needs to be transformed into a supervised learning problem. Additionally, evaluating the performance of an XGBoost model for time series forecasting requires specialized techniques, such as walk-forward validation, to obtain reliable and unbiased results[38].

In summary, the key characteristics of XGBoost are: - Efficient implementation of the gradient boosting ensemble algorithm. - Exceptional speed and efficiency, making it well-suited for various predictive modeling tasks. - High-performance track record in data science competitions and real-world applications. - Ability to handle time series forecasting, although requiring the transformation of data into a supervised learning problem[38].

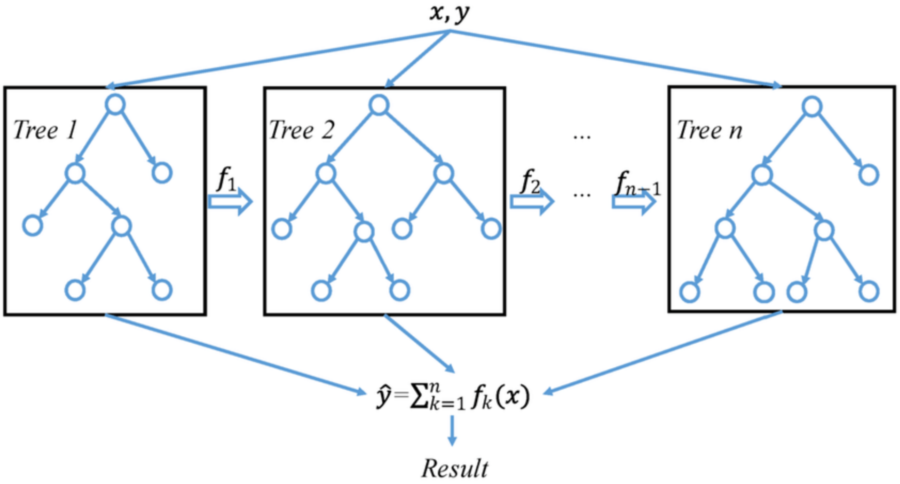

Figure 17. *A general architecture of XGBoost[39].*

2.3.6 LSTM

LSTM (Long Short-term Memory) networks are a variant of RNNs that address the vanishing gradient problem[1] and capture longer-term dependencies in time series data. More details about LSTMs can be found in the literature. LSTMs are characterized by an internal (hidden) state, denoted as h, and a cell state, denoted as c. The computation process of an LSTM involves three dependencies: the previous cell state c', the previous internal state h', and the input at the current time point x. This process incorporates the forget gate, input gate, addition gate, and output gate, which implement the functions f_t, i_t, $\tilde{C}_t$, and o_t, respectively, allowing for precise control over learning longer-term dependencies[1].

The computation process involved in an LSTM is depicted in The figure 18.

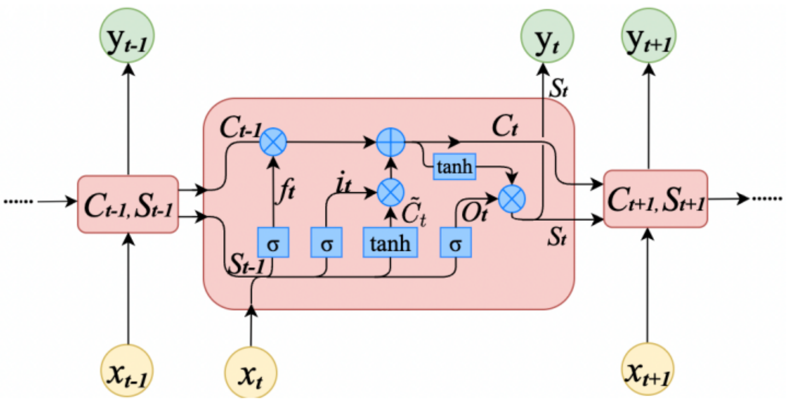

Figure 18. *Computation process involved in an LSTM[39].*

The equations governing the LSTM computation process are as follows:

$$\begin{aligned} i_t &= \sigma(W_i \cdot [S_{t-1}, x_t] + b_i), \\ f_t &= \sigma(W_f \cdot [S_{t-1}, x_t] + b_f), \\ \tilde{C}_t &= \tanh(W_C \cdot [S_{t-1}, x_t] + b_C), \\ C_t &= f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t, \\ o_t &= \sigma(W_o \cdot [S_{t-1}, x_t] + b_o), \\ S_t &= o_t \cdot \tanh(C_t) \end{aligned} \tag{2.10}$$

[1] The vanishing gradient problem in Recurrent Neural Networks (RNNs) refers to the phenomenon where the gradients diminish exponentially during backpropagation, leading to difficulties in learning long-term dependencies.

Here, x_t represents the input vector of d input features at time t, W_i, W_f, W_C, W_o are parameter matrices, b_i, b_f, b_C, b_o are bias vectors, and σ denotes the sigmoid activation function. The number of neurons in the LSTM layer is denoted as m.

LSTMs provide a powerful framework for modeling long-term dependencies in time series data, making them well-suited for tasks such as time series forecasting and sequence generation[1].

2.3.7 LSTM Variants

Various LSTM variants have emerged in the literature, each introducing specific modifications to enhance performance or address particular challenges. Some notable variants include :

Gated Recurrent Units

Gated Recurrent Units (GRU) is a variant of LSTMs to further address the vanishing gradient problem. As shown in Figure 19, the novelty in this method is in the use of an update gate, a reset gate, and a third gate, implementing the functions z_t, r_t, and $\tilde{S}_t$, respectively. Each gate plays a different role in controlling how to filter, use, and combine prior information. The first term in the expression for the next state, given by $(1-z_t)\cdot S_{t-1}$, decides what to retain from the past, while $z_t \cdot \tilde{S}_t$ determines what to collect from the current memory content [1].

The equations for the GRU computation process are as follows:

$$
\begin{aligned}
z_t &= \sigma(W_z \cdot (x_t \oplus S_{t-1}) + b_z), \\
r_t &= \sigma(W_r \cdot (x_t \oplus S_{t-1}) + b_r), \\
\tilde{S}_t &= \tanh(W_s \cdot (x_t \oplus (r_t \odot S_{t-1})) + b_s), \\
S_t &= (1 - z_t) \odot S_{t-1} + z_t \odot \tilde{S}_t, \\
y_t &= W_y \cdot S_t + b_y
\end{aligned}
\tag{2.11}
$$

where x_t is the input vector of n features at time t, W_z, W_r, W_s, W_y are parameter matrices, b_z, b_r, b_s, b_y are bias vectors, σ is the sigmoid function, and $\odot$ represents element-wise multiplication[1].

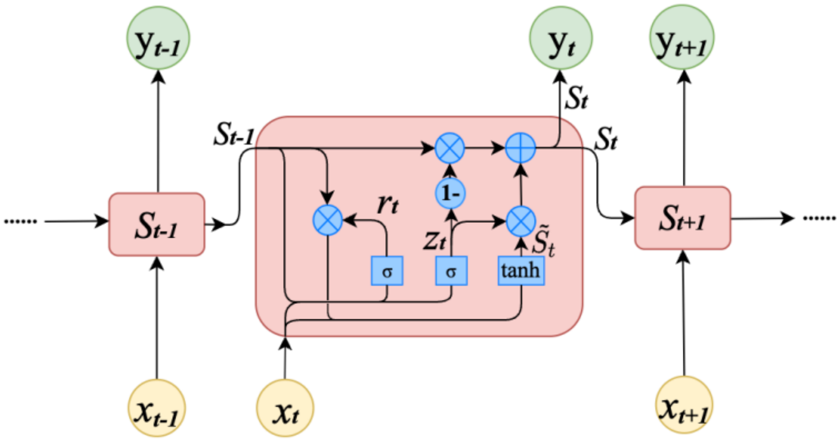

Figure 19. *Computation process of GRU[1].*

CNN-LSTM

The CNN Long Short-Term Memory Network (CNN LSTM) is an architecture designed specifically for sequence prediction tasks that involve spatial inputs like images or videos. It combines the strengths of Convolutional Neural Network (CNN) layers for extracting features from input data and Long Short-Term Memory (LSTM) units for modeling sequences.

The CNN LSTM architecture, also known as Long-term Recurrent Convolutional Networks (LRCN), has been primarily used for tasks such as activity recognition, image description, and video description. It leverages the deep spatial and temporal characteristics of CNNs and the sequential modeling capabilities of LSTMs. By using a pre-trained CNN as an image encoder and feeding its output into the LSTM decoder, the CNN LSTM can generate descriptive captions for images.

This architecture is suitable for problems that exhibit both spatial and temporal structures in the input data, such as the pixel arrangement in images or the word order in sentences. It is capable of capturing temporal dependencies and generating output with temporal structure. The CNN LSTM has proven effective in various tasks, including speech recognition and natural language processing, where CNNs are used to extract features from audio and textual inputs.

In summary, the CNN LSTM is a powerful solution for sequence prediction tasks involving spatial inputs. It combines the feature extraction capabilities of CNNs with the sequential modeling capabilities of LSTMs, making it suitable for tasks that require understanding both the spatial and temporal aspects of the input data[40].

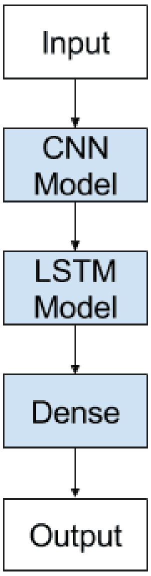

Figure 20. *Convolutional Neural Network Long Short-Term Memory Network Architecture[40].*

Bidirectional LSTM

Briefly, a Bidirectional LSTM (Long Short-Term Memory) is a type of recurrent neural network (RNN) architecture that processes input data in both forward and backward directions. It consists of two LSTM layers, one processing the input sequence in the forward direction and the other in the reverse direction. This allows the model to capture both past and future context, enabling it to make more accurate predictions and capture long-range dependencies in sequential data[41].

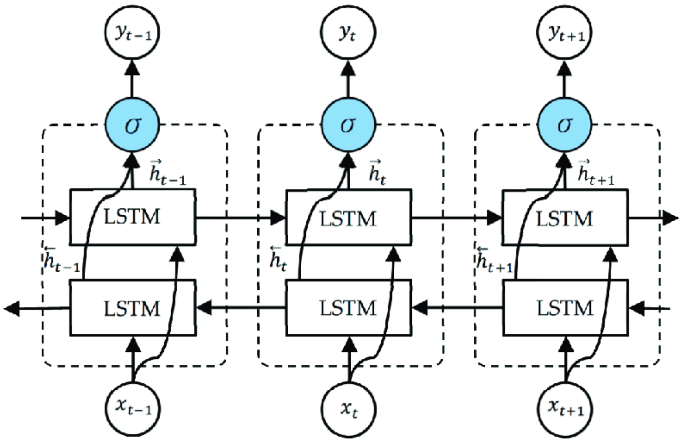

Figure 21. *The unfolded architecture of Bidirectional LSTM (BiLSTM) with three consecutive steps.[42].*

2.3.8 Reservoir Computing and Echo states

Reservoir Computing (RC) is an emerging technique in the field of recurrent neural networks that offers an efficient solution for training these networks. One of the key components of RC is the utilization of a large, untrained dynamical system called the reservoir. This reservoir can be excited with one or more inputs and serves as the foundation for the RC framework[43].

In RC, the desired output function is typically achieved through a linear memory-less mapping of the full instantaneous state of the reservoir. This means that only the linear mapping is learned using standard linear regression techniques, while the reservoir itself remains untrained. This approach offers several advantages, including the ability to overcome the vanishing gradient problem commonly encountered in training recurrent neural networks.

A notable concept within RC is the notion of echo states. These echo states refer to the reservoir's ability to retain information from past inputs and exhibit reverberating dynamics that influence the system's future behavior. By leveraging the echo states within the reservoir, RC can effectively capture complex temporal dependencies and facilitate accurate predictions.

Specifically, in the context of time series analysis, RC excels in tasks involving recursive prediction, where the generation of future time steps relies on a learned history. This makes RC particularly suitable for systems that incorporate delayed output feedback as an input to the reservoir. The utilization of echo states in such systems enables the efficient processing of sequential data and provides a powerful tool for time series prediction.

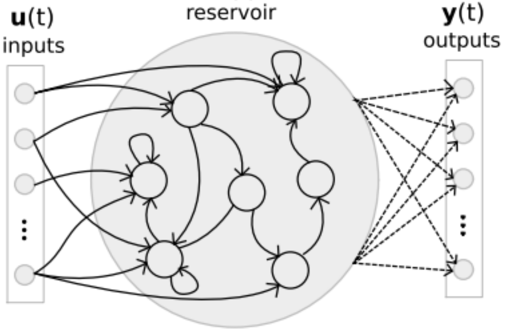

Figure 22. *Reservoir Computing network. The reservoir functions like a temporal kernel which projects the input to a rich feature space. Solid lines represent fixed connections. Dashed lines define connections which should be learned[44].*

By harnessing the power of echo states and the unique capabilities of reservoirs, RC has emerged as a promising approach for tackling complex sequential data analysis tasks, offering improved training efficiency and enhanced predictive performance.

2.3.9 Performance Metrics

In order to assess and compare the accuracy of the forecasting models for PV power, several performance metrics will be used. The following metrics are commonly employed in the evaluation of time series forecasting models:

Root Mean Squared Error (RMSE)

measures the average magnitude of the differences between the predicted and actual values:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{2.12}$$

Mean Absolute Error (MAE)

calculates the average absolute difference between the predicted and actual values:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{2.13}$$

Median Error

represents the median value of the differences between the predicted and actual values:

$$\text{Median Error} = \text{Median}(|y_i - \hat{y}_i|) \tag{2.14}$$

Coefficient of Determination (R^2 Score)

quantifies the proportion of the variance in the actual values that can be explained by the predicted values:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{2.15}$$

Where y_i represents the actual value, $\hat{y}_i$ denotes the predicted value, $\bar{y}$ is the mean of the actual values, and n is the total number of data points.

These metrics provide a comprehensive evaluation of the forecasting performance and will be used to compare the different models.

2.4 Hyperparameter Tuning with Meta-Heuristic search

2.4.1 Definition and Examples

In the field of machine learning, hyperparameter optimization or tuning refers to the task of selecting the most suitable hyperparameters for a learning algorithm. Hyperparameters are parameters that control the learning process, while other parameters, like node weights, are learned from the available data[45].

Different machine learning models often require specific hyperparameters to effectively capture diverse data patterns. The goal of hyperparameter optimization is to discover the optimal combination of hyperparameters that minimizes a predefined loss function when applied to independent data[45].

Here are some examples of hyperparameters for some of the machine learning models we have mentioned :

- Support Vector Regression (SVR):
  - Kernel type (e.g., linear, polynomial, radial basis function)
  - Regularization parameter (C)
  - Kernel coefficient (gamma)
- Decision Trees:
  - Maximum tree depth
  - Minimum number of samples required to split a node
  - Minimum impurity decrease for a split
- XGBoost:
  - Learning rate
  - Maximum tree depth
  - Number of estimators
- Random Forests:
  - Number of trees
  - Maximum tree depth

    - Minimum number of samples required to split a node
- Long Short-Term Memory (LSTM):
    - Number of LSTM units
    - Dropout rate
    - Learning rate
- Echo States:
    - Reservoir size
    - Spectral radius
    - Leakage rate

2.4.2 Classical Hyperparameter Optimization algorithms

Grid Search

Grid search is a classical hyperparameter optimization algorithm widely used in machine learning. It involves exhaustively searching through a specified subset of the hyperparameter space, typically by discretizing and bounding the parameter values. The algorithm evaluates the performance of the learning algorithm for each combination of hyperparameters using cross-validation or a hold-out validation set. Grid search outputs the hyperparameter settings that yield the highest performance score[45].

Random Search

Random search is an alternative hyperparameter optimization algorithm that selects hyperparameter combinations randomly instead of exhaustively. It is applicable to both discrete and continuous hyperparameter spaces and allows the inclusion of prior knowledge through specified distributions. Random search is embarrassingly parallel and serves as a baseline for evaluating the performance of new optimization methods[45].

These classical hyperparameter optimization algorithms, grid search and random search, offer different strategies for finding optimal hyperparameter settings. However, they suffer from limitations such as the curse of dimensionality. As machine learning models and datasets become more complex, advanced techniques like metaheuristic search algorithms are needed to overcome these limitations and efficiently explore high-dimensional and non-convex hyperparameter spaces.

2.4.3 Metaheuristic Search

In the domain of computer science and mathematical optimization, a *metaheuristic* refers to an advanced procedure or heuristic designed to explore, generate, fine-tune, or select heuristics (partial search algorithms) that can provide reasonably good solutions to optimization or machine learning problems. Metaheuristics are particularly useful when dealing with incomplete or imperfect information or limited computational capacity.

Metaheuristics sample a subset of solutions from an overwhelmingly large search space that cannot be exhaustively explored or enumerated. Unlike traditional optimization algorithms and iterative methods, metaheuristics do not guarantee finding globally optimal solutions for specific problem classes. Many metaheuristics utilize stochastic optimization techniques, where the resulting solution depends on a set of random variables generated during the search process[46].

In combinatorial optimization, metaheuristics often outperform optimization algorithms, iterative methods, or basic heuristics in terms of discovering good solutions with reduced computational effort. As a result, metaheuristics have become valuable approaches for solving optimization problems.

Although the literature on metaheuristics primarily focuses on empirical results obtained from computer experiments, some formal theoretical results exist, particularly concerning convergence properties and the potential for finding global optima[47][48].

While the field includes high-quality research, it is important to note that some publications lack clarity, conceptual elaboration, rigorous experiments, and awareness of prior literature.

Numerous classes of metaheuristic algorithms have been proposed, each claiming novelty and practical efficacy. The literature includes various books and survey papers that cover both the empirical aspects of metaheuristics based on computer experiments and the theoretical investigations into convergence properties and the potential for achieving global optima[46] [49].

In this research, we will investigate the effectiveness of the Differential Evolution (DE) and Particle Swarm Optimization (PSO) metaheuristic algorithms for tuning the proposed PV power forecasting models. Through a series of experiments, we aim to enhance the accuracy and reliability of the forecasts, providing valuable insights into the applicability of DE and PSO in this context.

2.4.4 Differential Evolution

Basic Description of DE

Differential Evolution (DE) is a metaheuristic stochastic algorithm proposed by Storn and Price (1997) that is both simple and effective. As an evolutionary algorithm, DE operates on a population of candidate solutions and iteratively improves them to find an optimal solution. DE stands out among other evolutionary genetic algorithms due to its simplicity, ease of implementation, and competitive performance in solving complex optimization problems[50][3].

DE is particularly suitable for derivative-free global optimization, where the objective function might be non-differentiable or noisy. It doesn't rely on gradient information, making it robust and applicable in a wide range of real-world problems. The algorithm's effectiveness lies in its ability to explore and exploit the search space efficiently, combining mutation and crossover operations to generate offspring with improved fitness[50].

The standard DE process involves the following main operations:

(a) Initialization: In this step, the algorithm initializes the following parameters: chromosome length D, gene value range $[U_{\text{min}}, U_{\text{max}}]$, population size N, crossover rate CR, and scaling or mutation factor F. The population is randomly initialized using the equation[3]:

$$x_{ij} = U_{\text{min}} + \text{rand} \cdot (U_{\text{max}} - U_{\text{min}}) \tag{2.16}$$

where $i = 1, 2, \ldots, N$, $j = 1, 2, \ldots, D$, and rand is a randomly generated value with a uniform probability distribution.

(b) Mutation: In this step, a mutated individual is generated using the equation:

$$v_{ij} = x_{r1} + F \cdot (x_{r2} - x_{r3}) \tag{2.17}$$

where r_1, r_2, and r_3 are randomly generated values that are distinct from each other and from the objective individual number i. The scaling factor F is distributed within the interval $[0, 2]$ as suggested in the literature.

(c) Crossover: The crossover operation generates an experimental individual using the equation:

$$u_{ij} = \begin{cases} v_{ij}, & \text{if rand}() \leq CR \text{ or randn}() = j \\ x_{ij}, & \text{otherwise} \end{cases} \tag{2.18}$$

where j represents the j-th gene of an individual, rand(j) is a random value in the uniform distribution $[0, 1]$, and CR is distributed within the interval $[0, 1]$. The randomly generated number randn$(t) \in [1, 2, \ldots, D]$ ensures that at least one dimension of the experimental individual comes from the mutated individual[3].

(d) Selection: The selection operation generates the offspring if the fitness value of the experimental individual is better; otherwise, the objective individual directly becomes the offspring. The equation is presented as follows, where f is the fitness function:

$$x_{i+1} = \begin{cases} u_i, & \text{if } f(u_i) \leq f(x_i) \\ x_i, & \text{otherwise} \end{cases} \tag{2.19}$$

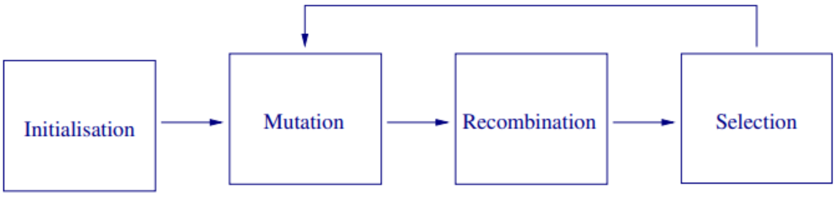

Figure 23. *General Evolutionary Algorithm Procedure.*

Parameter Selection

In the context of Differential Evolution (DE), the selection of appropriate parameter values significantly influences the outcome and convergence speed of the algorithm. In this study, the parameter values commonly employed by the research community, namely F=0.8 and C=0.9, were adopted. These values have been widely utilized due to their favorable properties and empirical effectiveness in addressing a variety of optimization problems[50].

The value of F, referred to as the scaling or mutation factor, determines the amplification of the differential variation between candidate solutions during the mutation operation. A value of 0.8 strikes a balance between exploration and exploitation, allowing for effective exploration of the search space while avoiding premature convergence. Empirical evidence suggests that this value often yields satisfactory results across a range of optimization scenarios.

On the other hand, C denotes the crossover rate, which governs the probability of recombining genetic information from the mutated candidate solution with the target individual. A value of 0.9 indicates a high likelihood of successful recombination, leading to efficient exploitation of promising regions in the search space. This value has been found to strike a good balance between diversification and intensification, facilitating both global and local search capabilities.

While the selection of parameter values may vary depending on the specific problem and problem characteristics, the choice of F=0.8 and C=0.9 in this thesis is justified based on their established effectiveness and widespread usage within the research community. These values offer a reasonable starting point for DE optimization tasks, providing a foundation for further investigation and customization as needed in specific applications.

The selection of an appropriate population size is a crucial aspect in Differential Evolution (DE). A careful choice is required to avoid computational inefficiency associated with large population sizes or suboptimal solutions resulting from small population sizes. In this study, we adhered to the widely accepted practice of using a population size of 10N, where N denotes the dimensionality of the search space, to strike a balance between solution quality and computational efficiency. By aligning with this recommended value, we established a baseline for performance comparison and conducted additional experiments to investigate the impact of population size variations on convergence behavior and solution quality. This rigorous approach ensures the robustness and validity of our findings concerning the significance of population size selection in DE.

DE variants

DE (Differential Evolution) encompasses numerous variants that differ in their mutation strategies. Some of the prominent variants include the following:

- Rand/1/bin[2]: The target vector is mutated by adding the difference between two randomly selected individuals from the population, scaled by a mutation factor.
- Rand/2/bin: The target vector is mutated using the differences between three randomly selected individuals.

[2] In the ’bin’ strategy, binomial crossover is used.

- Best/2/bin: The target vector is mutated by combining the best individual with two randomly selected individuals from the population.
- Rand-to-best/1/bin: The target vector is mutated by adding the difference between the best individual and a randomly selected individual to the mutation of two other individuals.
- Best/2/exp[3]: The target vector is mutated by combining the best individual with two exponential crossover operations.

A study conducted by G. Jeyakumar and C. Shanmugavelayutham compared the convergence speed of these different strategies. The results suggested that the best performing variants (rand/1/bin, rand/2/bin, best/2/bin, rand-to-best/1/bin, and best/2/exp) exhibited faster convergence towards the solution[51].

In our study, we utilized two specific strategies, namely rand/1/bin and best/1/bin. The rand/1/bin strategy involves mutating the target vector by adding the difference between two randomly selected individuals to the target vector itself. The equation for rand/1/bin mutation is given by:

$$v_i = x_{r1} + F \cdot (x_{r2} - x_{r3}) \tag{2.20}$$

where v_i is the mutated target vector, x_{r1}, x_{r2}, and x_{r3} are randomly selected individuals from the population, and F is the mutation factor.

On the other hand, the best/1/bin strategy mutates the target vector by adding the difference between the best individual and a randomly selected individual. The equation for best/1/bin mutation is given by:

$$v_i = x_{\text{best}} + F \cdot (x_{r1} - x_i) \tag{2.21}$$

where x_{best} is the best individual in this case.

It is worth noting that there exist other variants of DE that employ reduction population strategies to minimize the number of function evaluations and incorporate parameter adaptive selection for mutation and recombination.

[3] In the 'exp' strategy, exponential crossover is used.

In our study, we employed the traditional DE implementation provided by the SciPy library in Python to select the hyper-parameters which minimize the Mean Squared Error.

2.4.5 Particle Swarm Optimization

Basic Description

Particle Swarm Optimization (PSO), proposed by Kennedy and Eberhart in 1995, is a computational method that iteratively improves candidate solutions to optimize a given problem based on a measure of quality. Inspired by sociobiological concepts, PSO simulates the behavior of a flock of birds or a school of fish, where the collective knowledge and experience of all members benefit the group as a whole. In PSO, each candidate solution is represented as a particle in a high-dimensional solution space. The particles move within the search-space by adjusting their positions and velocities according to simple mathematical formulas. The movement of each particle is influenced by its own best-known position and the best-known positions of other particles in the swarm. By continuously updating the best-known positions, the swarm is guided towards better solutions[52][53].

PSO is a metaheuristic optimization algorithm that does not rely on specific problem assumptions and can explore large solution spaces effectively. Unlike traditional optimization methods that require differentiability, PSO does not use gradient information, making it suitable for non-differentiable problems. Although PSO cannot guarantee finding the true global optimum, empirical evidence suggests that PSO often converges to near-optimal solutions. The theoretical foundations and applications of PSO have been extensively studied, highlighting its effectiveness as a powerful optimization technique[52][53].

Algorithm Details

The algorithm involves the following components:

- Particle Position: Each particle in the swarm is assigned a position vector, denoted as X_i.
- Particle Velocity: Each particle also has a velocity vector, denoted as V_i.
- Best Individual Particle Position: The best position found by an individual particle is denoted as P_i.
- Best Swarm Position: The best position found by the entire swarm is denoted as P_{best}.

The PSO algorithm iteratively updates the particle positions and velocities based on the following equations:

$$V_i(k+1) = wV_i(k) + c_1 r_p(P_i(k) - X_i(k)) + c_2 r_g(P_{\text{best}}(k) - X_i(k)) \tag{2.22}$$

$$X_i(k+1) = X_i(k) + V_i(k+1) \tag{2.23}$$

where:

- w is the inertia weight
- c_1 and c_2 are cognitive and social parameters
- r_p and r_g are random numbers between 0 and 1

The PSO algorithm follows the following flow diagram:

1. **Initialize**
   - Set constants w, c_1, and c_2 to predefined values.
   - Randomly initialize particle positions X_i in the solution space for $i = 1, \ldots, p$.
   - Randomly initialize particle velocities V_i for $i = 1, \ldots, p$.
   - Set iteration counter $k = 1$.
   - Set maximum number of iterations K.
2. **Optimize**
   - Evaluate the objective function value f using the design space coordinates.
   - If $f(P_i) < f(P_{\text{best}})$, update $P_{\text{best}} = P_i$.
   - If the stopping condition or maximum number of iterations K is reached, go to step 3.
   - Update all particle velocities V_i for $i = 1, \ldots, p$ using the velocity equation.
   - Update all particle positions X_i for $i = 1, \ldots, p$ using the position equation.
   - Increment k.
   - Go to step 2.
3. **Terminate**

The PSO algorithm aims to find the optimal solution by iteratively updating the positions and velocities of particles in the swarm. The inertia weight w controls the trade-off between exploration and exploitation. The cognitive parameter c_1 emphasizes the particle's own best position, while the social parameter c_2 guides the particle towards the swarm's best position. The algorithm terminates either when a stopping condition is met or the maximum number of iterations K is reached[53][52].

PSO variants

Particle Swarm Optimization (PSO) offers various possibilities for customization and improvement. Different approaches exist for initializing particles and velocities, velocity damping, and updating personal best positions and global best positions. These choices have been extensively studied in the literature to understand their impact on algorithm performance. In order to establish a standardized framework for evaluating advancements in PSO, leading researchers have developed standard implementations of the algorithm. These standardized versions, such as the widely recognized SPSO-2011, serve as benchmarks for comparing and testing new approaches in the field of optimization[53].

Our thesis employed the conventional PSO algorithm using the "pyswarm" library in Python to select the hyper-parameters which minimize the Mean Squared Error. This allowed us to focus on our research objectives by utilizing a reliable and standardized implementation. Leveraging the capabilities of PSO facilitated effective investigation and attainment of our research goals.

2.4.6 Cross Validation For time series

In addition to optimization strategies, evaluating a proposed model for potential overfitting or underfitting is a central concern in Data science. To address this, cross-validation strategies are essential in assessing the model's performance[34].

Cross-validation involves partitioning the dataset into training, validation, and withhold sets, allowing the construction and testing of the model. Overfitting occurs when increasing the model complexity or training epochs improves performance on the training set but leads to higher error on the withhold set. Conversely, underfitting arises when the model fails to capture the underlying patterns adequately. Cross-validation serves as a crucial tool to mitigate these challenges and refine hyperparameters for optimal model performance[34].

K-fold Cross-Validation

K-fold cross-validation is a widely used technique for evaluating the performance of a model and assessing its generalization ability. In this process, the dataset is divided into K folds or partitions, with one fold held as a test set and the remaining K-1 folds used for training. The model is trained and evaluated K times, each time using a different fold as the test set. The results are then averaged or combined to obtain a more robust estimate of the model's performance. This approach helps to gauge the model's effectiveness, especially in terms of its ability to generalize to new, unseen data[34].

Traditional k-fold cross-validation is unsuitable for evaluating models on time series data due to the temporal nature of the data. Random partitioning in k-fold cross-validation ignores the temporal structure and can lead to over-optimistic performance estimates. Instead, techniques like time series cross-validation or rolling window validation preserve the temporal order, providing more reliable performance evaluation for time series analysis[54].

Time series Cross-Validation

Time series split is a specialized technique for assessing models on time series data. It avoids the pitfalls of traditional cross-validation by maintaining the temporal order of observations. The data is sequentially divided into training and testing sets, with the training set containing earlier time periods and the testing set containing later ones. This approach ensures realistic evaluation, considering the inherent temporal dependencies in the data. By employing time series split, accurate performance assessment of models in time series analysis can be achieved.

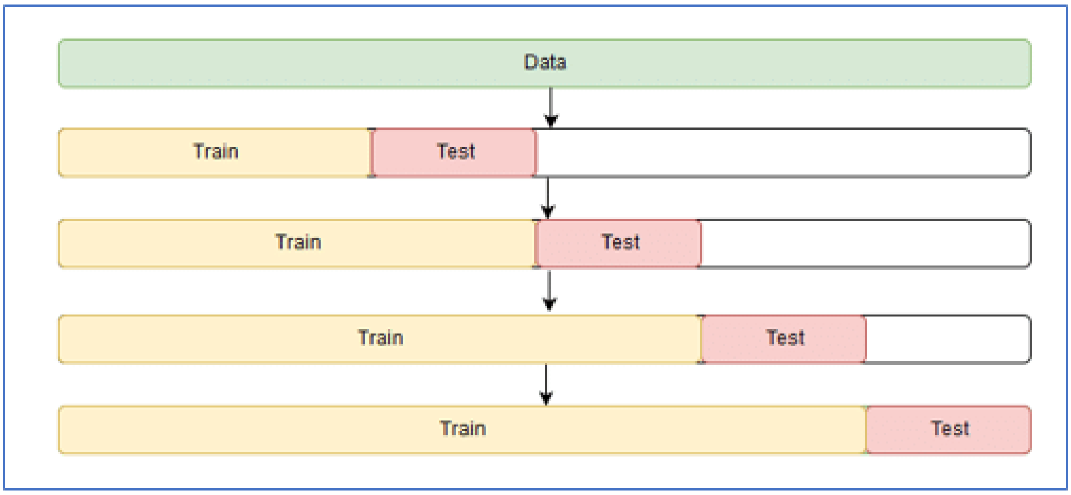

Figure 24. *Time Series Cross Validation[55].*

In our study, we conducted a rigorous hyperparameter optimization process by employing a three-fold time series split. This approach, implemented using the time series split function in Python's sklearn library, allowed us to robustly evaluate our models while considering the temporal dependencies within the data.

2.5 Conclusion

In conclusion, this chapter provided a comprehensive overview of time series analysis and machine learning models for PV power forecasting. We began by introducing the fundamentals of time series analysis, including its definition, basic characteristics, and forecasting techniques. We then explored the limitations of traditional statistical models in time series forecasting and the emergence of machine learning approaches.

The chapter further delved into machine learning for time series forecasting, discussing different paradigms and emphasizing the regression approach for PV power forecasting. Various machine learning models suitable for PV power forecasting were presented, including linear regression, support vector machines, decision tree regression, random forests, XGBoost, LSTM, LSTM variants, reservoir computing, and echo states.

Additionally, we covered the importance of performance metrics in evaluating the accuracy of the forecasting models. Finally, we discussed hyperparameter tuning using metaheuristic search techniques, such as differential evolution and particle swarm optimization, and highlighted the significance of cross-validation for time series data.

Overall, this chapter provided a solid foundation for understanding time series analysis and machine learning models in the context of PV power forecasting. It equips researchers and practitioners with the necessary knowledge to apply these techniques effectively and make informed decisions for accurate power forecasting in the renewable energy sector.

3. Data Analysis and Feature Engineering

Feature engineering is a crucial step in predictive modeling using machine learning, as it allows us to transform raw data into meaningful features that capture underlying patterns and relationships. By carefully selecting and engineering informative features, we enhance accuracy and performance. In this chapter, we discuss dataset characteristics, data preprocessing techniques and feature selection methods. By empowering machine learning algorithms to learn meaningful patterns, we make more accurate predictions and drive innovation in various industries.

3.1 Data Presentation

we will start by discussing the sites where we get the datasets used in our project. Having good quality and diverse datasets is crucial for developing accurate and reliable forecasting models. For our study, we used two different datasets: one from DKASC in Australia and the other from the Dhaya Power Plant in Algeria. Each dataset has unique features that contribute to the thorough analysis and evaluation of our forecasting models. Throughout this section, we will explore the characteristics of these photovoltaic sites, how we collected the data, and the weather data we gathered.

3.1.1 The Desert Knowledge Australia Solar Center

The Desert Knowledge Australia Solar Center (DKASC) is the largest solar demonstration facility in the southern hemisphere, located in Alice Springs, Australia. It integrates research and commercial applications, attracting researchers and visitors worldwide. The arid desert climate of Alice Springs provides abundant solar resources for DKASC, which houses 38 sites demonstrating various photovoltaic (PV) technologies. The facility serves as a platform for studying and evaluating different PV systems. DKASC promotes renewable energy, drives innovation in solar power, and contributes to the global adoption of sustainable solar technologies.Figure 25 represents the distribution of PV systems in DKASC:

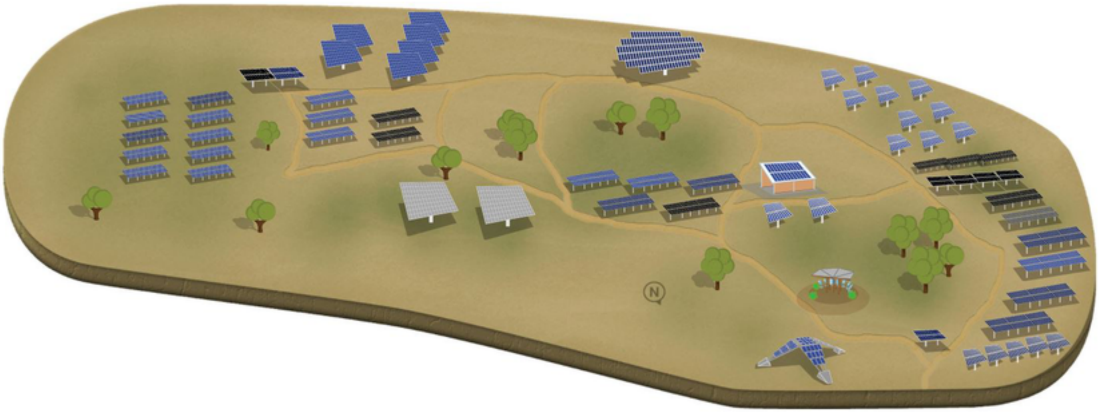

Figure 25. *Representative chart of the DKASC PV plant [56].*

The Sanyo solar panels

The study conducted at the Desert Knowledge Australia Solar Center (DKASC) focuses on a particular array that utilizes Sanyo HIT (Heterojunction with Intrinsic Thin Layer) hybrid silicon technology. This ground-mounted array is tilted at an angle of 20 degrees to optimize solar energy capture.

It is worth noting that this specific array, which has been in operation since its installation on March 11, 2010, is depicted in figure 26.

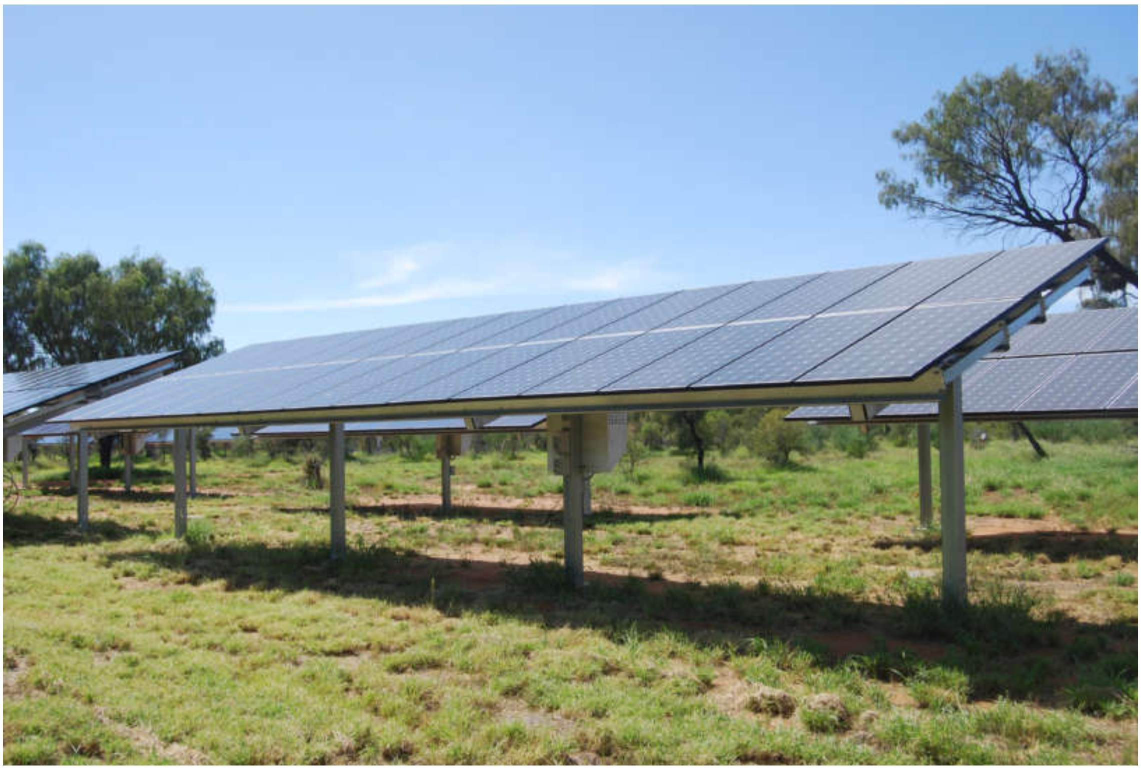

Figure 26. *the Sanyo PV panels[57].*

Some characteristic of the panel are shown in table 1:

Property	**Value**
Array Rating	6.3 kW
Panel Rating	210 W
Number of Panels	30
Panel Type	Sanyo HIP-210NKHE5
Array Area	37.83 m²
Inverter Size / Type	7 kW, SMA SMC 7000TL

Table 1. characteristic of the Sanyo panel

The array's integration of Sanyo HIT hybrid silicon technology, its dimensions, orientation, and the associated SMA inverter demonstrate the specific configuration and components employed in this study at the DKASC.

The weather data

To gather data for research purposes, the DKASC relies on high-resolution sensors that record data every 5 minutes. This data encompasses various weather parameters such as temperature, relative humidity, global and diffuse radiation, wind direction, and radiation on tilted surfaces. This information is used for numerical validation in research studies.[58]

- The dataset utilized for this study spans from January 01, 2020, to December 31, 2022, providing an extensive range of information for training and validation purposes. With a total of 315,648 samples, the dataset offers a robust foundation for analysis.

figure 27 shows the dataset plotted:

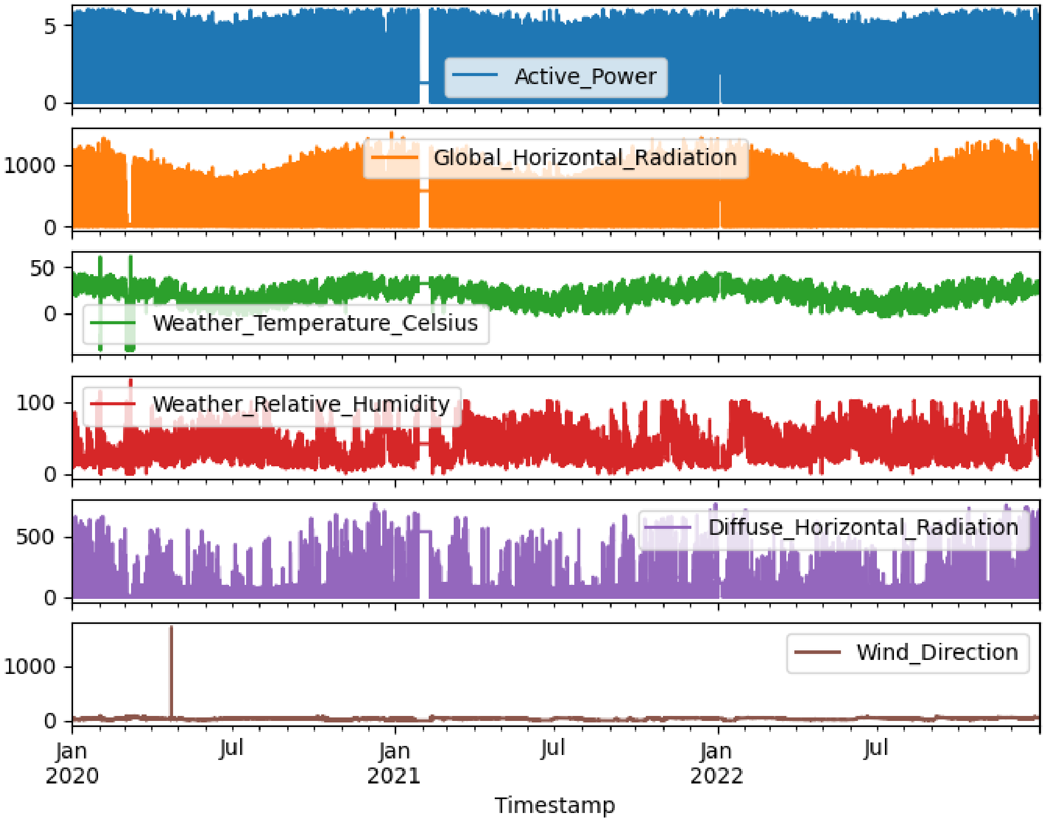

Figure 27. *DKASC data plotted.*

3.1.2 The Dhaya PV power plant

Established in 2016, the Dhaya PV plant emerged as a result of a collaborative effort between BELECTRIC Group, a German operator, and Shariket Kahraba wa Taket Moutadjadida (SKTM), an Algerian electricity production company. The Dhaya Photovoltaic (PV) power plant is situated in the Dhaya administrative district, located in the southernmost part of the Sidi Bel Abbés province in Algeria. It is positioned approximately 73 km away from the provincial capital. Spanning over an area of 32.6 hectares, the construction of this plant incurred an estimated cost of 21 million euros. Boasting a capacity of 12 MW, it contributes to the national renewable energy program by directly injecting the generated power into the national 60 kV grid. The project aligns with the country's endeavor to foster renewable energy sources for sustainable economic development.The overall diagram of this power plant is illustrated in figure 28.

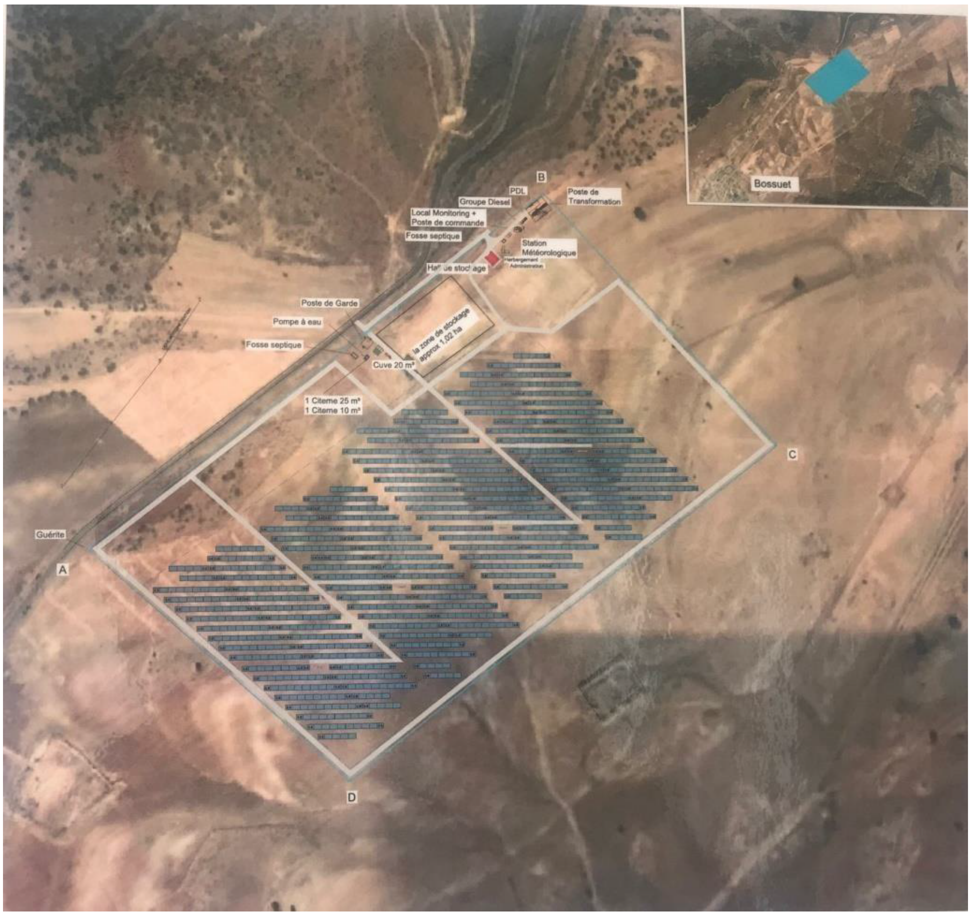

Figure 28. The general diagram of the Dhaya photovoltaic power plant in Sidi Bel Abbés*[59]*.

Solar panels characteristics at Dhaya power plant

The PV solar fields at the Dhaya plant feature 47,808 photovoltaic panels, specifically the HSL60P6-PB-1-250 polycrystalline silicon type. Each panel integrates 60 series-connected cells, providing a peak power of 250 Wp. The panels are oriented towards the south at a tilt angle of 15 degrees.

Table 2 present the relevant parameters:

Parameter	**Value**
Peak Power (Wp)	250W
Optimum Operating Voltage (Vmp)	30.4V
Optimum Operating Current (Imp)	8.23A
Open-Circuit Voltage (Voc)	37.7V
Short-Circuit Current (Isc)	8.79A
Maximum System Voltage	1000V

Table 2. parameters of the panel

The weather data

The Dhaya PV plant incorporates various environmental measurement devices in addition to power-related data. These devices include the following:

- ISO9060 pyrometer: This device is employed for the purpose of measuring solar radiation.
- DLE120 temperature sensors: These sensors are utilized to measure temperature.
- DMA672.1 humidity sensor: This sensor is employed to measure humidity.
- KIT 5.0 anemometer: This device is used to measure wind speed.
- Pluviometer: This instrument measures rainfall.

The primary objective of these devices is to collect environmental data. Subsequently, this data is recorded and stored in the data acquisition computers that are installed within the control room.figure 29 illustrates the Meteorological mini-station of the Dhaya power plant.

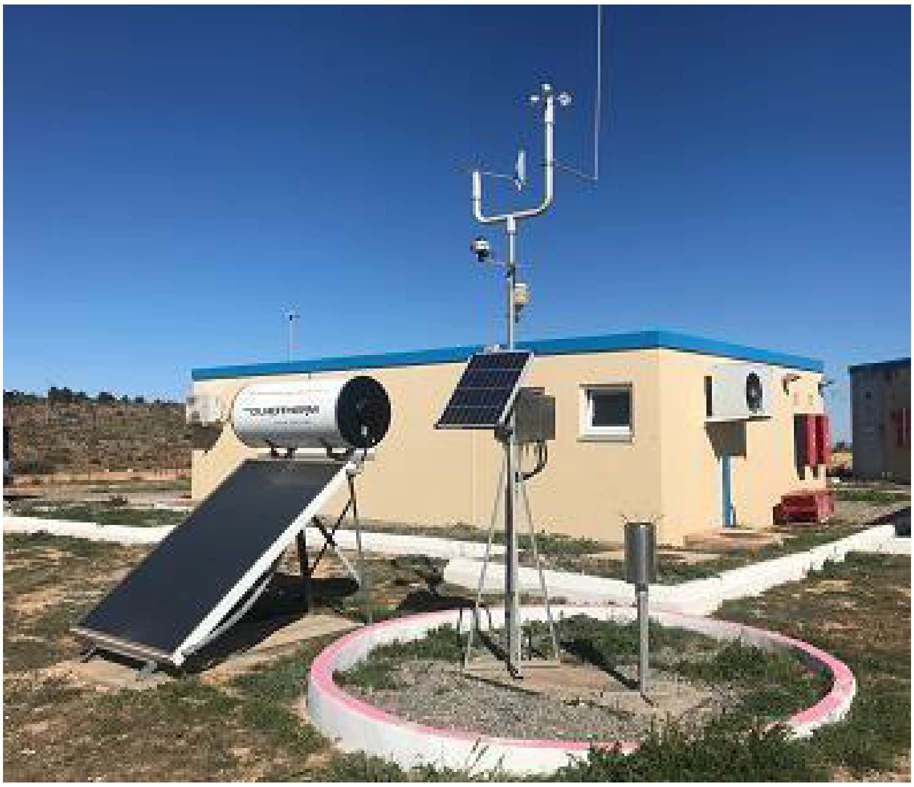

Figure 29. Meteorological mini-station of the Dhaya power plant*[59]*.

Specifically, at the Dhaya PV plant, the data collected from the selected skid (referred to as skid 1) encompasses power (W), solar radiation (G), ambient temperature (Ta), module temperature (Tm), and wind speed (v). These parameters are recorded at regular

15-minute intervals and stored in one of the data acquisition computers located in the control room.[59]

- The dataset utilized in our research was acquired from the "Centre de Développement d'Énergie Renouvelable" (CDER), a renowned institution in Algeria focused on the advancement of renewable energy technologies. The dataset obtained from CDER encompassed the essential information required for our project.
- The dataset utilized for this study spans from January 01, 2018, to 13 November, 2022, but includes only the daytime data. With a total of 82,848 samples, the dataset offers a robust foundation for analysis.

figure 30 shows the dataset plotted:

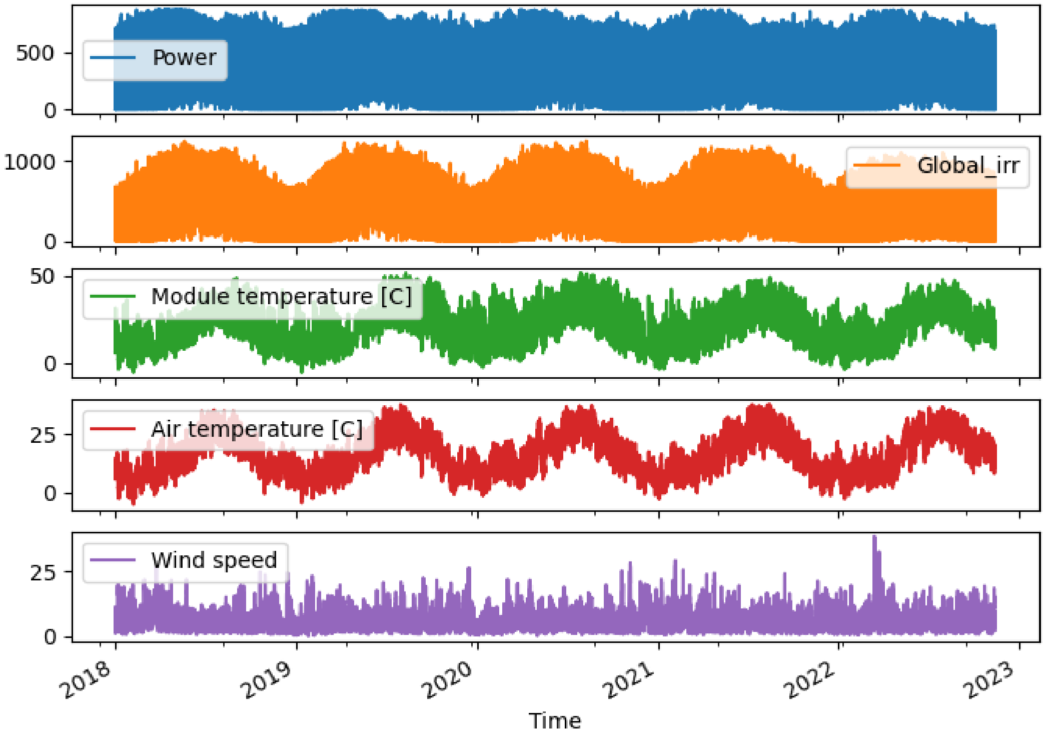

Figure 30. *Dhaya PV plant data plotted.*

3.2 Data Preprocessing

this section on data preprocessing delves into the practical aspects of preparing raw data for analysis. It covers various techniques and tools employed to clean, transform, and organize data, ensuring its suitability for further analysis.

3.2.1 Handling Missing Values

When examining both data sets using a Python program, we observed the presence of certain missing values. Addressing these missing values is crucial to enhance the performance of the forecasting algorithms.

As depicted in the previous data plotted graphs, both data sets exhibit missing values. To rectify this issue, there exist various approaches, including filling, ignoring, and interpolation...etc.

For small data gaps, we utilized the pandas backward filling operation, which involves filling the missing values with the preceding known values. This approach proves effective in cases where the temporal order of the data plays a significant role, as it ensures a smooth continuation of values.

To address the issue of missing values for days, weeks, or months, we implemented a strategy that involved utilizing data from adjacent time periods in the previous or subsequent year. By taking advantage of the natural seasonality observed in weather patterns across different years, we aimed to incorporate comparable weather conditions for these missing periods. This approach acknowledges the cyclic nature of weather patterns and contributes to preserving the overall consistency and continuity of the dataset.

Other approaches, such as ignoring the missing values, were deemed unsuitable for our analysis. Ignoring missing values could introduce noise into the model and hinder the system's ability to discern meaningful patterns. Thus, we opted to address the missing values using appropriate techniques rather than omitting them entirely.

3.2.2 Handling Outliers

Once we have addressed the issue of missing data, we can proceed to tackle outliers within our dataset. By organizing the data into a tabular format shown in table 3 and 4,we can identify values that fall outside the expected range. These data points are referred to as outliers, and their presence can adversely affect the performance of our machine learning model.

Variable	Count	Mean	Std	Min	25%	50%	75%	Max
Active_Power	315648	1.46	1.98	-0.00	0.00	0.00	3.06	6.05
Diffuse_Radiation	315648	62.84	108.79	0.00	1.32	8.89	78.55	769.86
Global_Radiation	315648	271.86	365.92	0.00	2.12	12.20	566.89	1524.54
Humidity	315648	37.25	22.76	0.00	19.29	31.62	51.23	131.16
Temperature	315648	20.81	10.70	-39.99	14.59	21.57	28.07	61.92
Wind_Direction	315648	29.91	15.73	-6.98	17.43	34.20	41.27	1701.37

Table 3. Summary statistics for DKASC Dataset

Variables	count	mean	std	min	25%	50%	75%	max
Power	82848	390.31	268.99	0.00	130.88	393.23	641.84	881.82
Global_irr	82848	454.57	326.67	0.00	149.92	432.44	724.10	1,247.12
Module temperature	82848	22.02	11.16	-5.80	13.01	22.22	30.57	51.70
Air temperature	82848	17.14	8.88	-5.28	9.92	16.52	24.37	37.57
Wind speed	82848	6.74	3.61	0.08	4.29	6.01	8.26	38.30

Table 4. Summary statistics for Dhaya Dataset

Outliers can be caused by various factors such as measurement errors, data corruption, extreme events, or rare occurrences. They violate assumptions and can cause model instability, resulting in unreliable forecasts with larger errors. As a result, it is crucial to handle outliers appropriately to ensure the reliability and robustness of our forecasting model.

To enhance our understanding of the variable distributions, we utilize box plots.

Box plot

A box plot, also known as a box-and-whisker plot, is a graphical representation of the distribution of a dataset. It provides a summary of key statistical measures and allows for a visual comparison between different groups or variables. The plot consists of several elements that represent various aspects of the data.

Here is a breakdown of the components of a typical box plot:

1. Median (Q2): The median represents the middle value of the dataset when it is ordered from smallest to largest. It divides the data into two equal halves, with 50%

of the values falling below and 50% above it. In the box plot, the median is depicted as a horizontal line inside a box.

2. Quartiles (Q1 and Q3): Quartiles are values that divide the dataset into four equal parts, each containing 25% of the data. The first quartile (Q1) marks the point below which 25% of the values lie, while the third quartile (Q3) represents the point below which 75% of the values lie. In the box plot, Q1 and Q3 are shown as the lower and upper edges of the box, respectively.
3. Interquartile Range (IQR): The IQR is a measure of the spread of the data and is calculated as the difference between the third quartile (Q3) and the first quartile (Q1). It covers the middle 50% of the dataset. In the box plot, the box is drawn to represent the IQR.
4. Minimum and Maximum: The minimum and maximum values represent the smallest and largest observations in the dataset, respectively. They define the range within which the majority of the data points lie. In a box plot, the whiskers extend up to these minimum and maximum values unless there are outliers present, in which case the whiskers extend to the furthest non-outlier data points within a certain range.
5. Whiskers: The whiskers extend from the box and represent the range of the data. They typically extend up to a certain distance from the box, typically 1.5 times the IQR. Observations beyond the whiskers are considered outliers and are represented as individual points.[60]

Here are the box plots generated for our datasets:

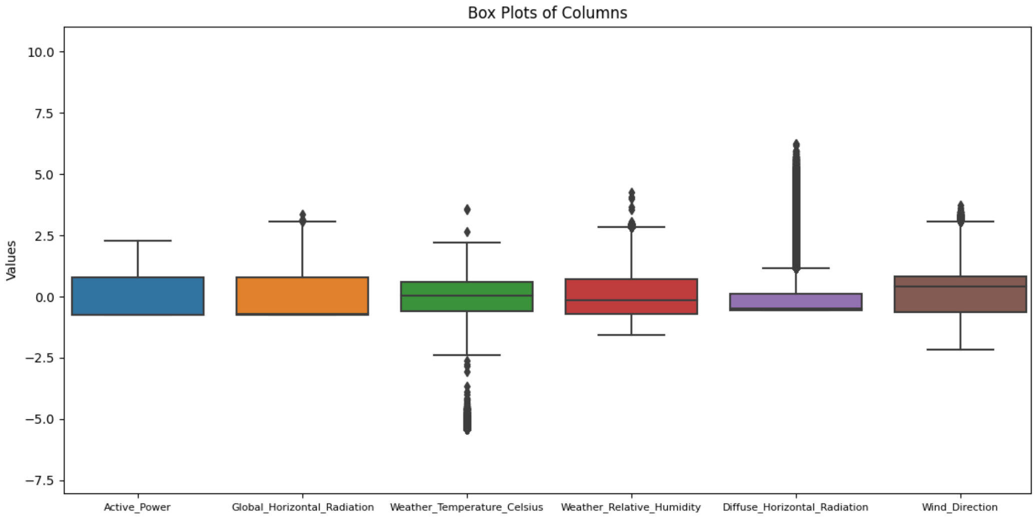

Figure 31. *DKASC data box plot.*

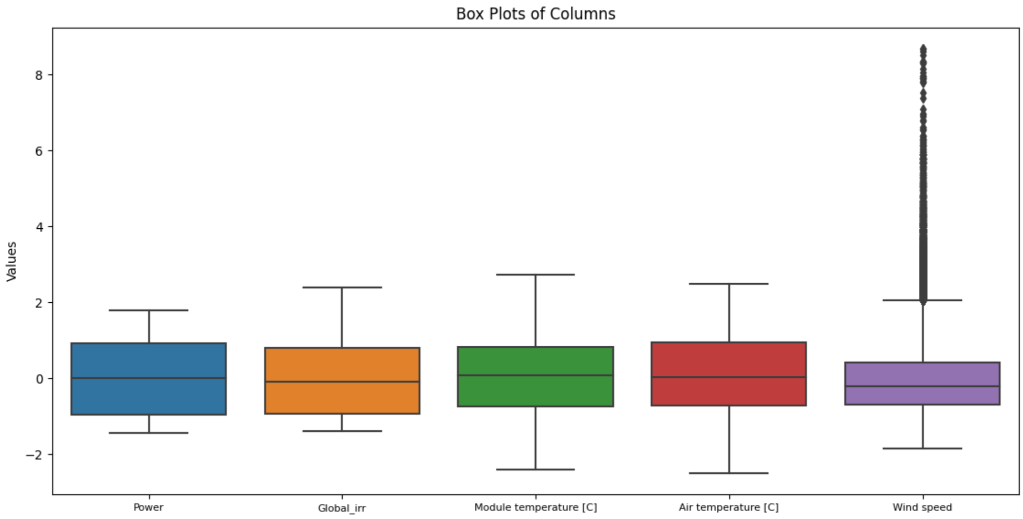

Figure 32. *Dhaya PV plant data box plot.*

By examining the box plots, we can identify outliers that lie beyond the whiskers of the plot, indicating extreme values that deviate significantly from the norm.

To mitigate the impact of outliers, we adopt a strategy that involves assigning them the maximum or minimum values observed in the dataset, depending on their position relative to the box plot. This approach effectively brings the outliers back within the range of the majority of the data, minimizing their influence on subsequent analyses.

3.2.3 Cycling coding

In the context of analyzing time series data using machine learning techniques, incorporating temporal features is a widely adopted approach. When dealing with data like irradiance, it exhibits a distinct characteristic of periodicity. For instance, the irradiance data consistently shows high values at noon. By considering this periodic nature during the training of machine learning models, we can significantly improve the accuracy of forecasting models.

Temporal data, represented by month, day, hour, and minute, follows a sequential format. However, the inherent periodicity of sequential data is not effectively captured by traditional machine learning models. For example, in the sequential format, the time points of 23 o'clock and 0 o'clock are treated as numerically distant, despite being adjacent in reality. To adequately represent the periodicity of temporal data, we introduce the following "sin/cos encoding" equations. These equations transform the sequence data from its original one-dimensional space to a continuous two-dimensional space, enhancing its periodic characteristics[61]:

$$\text{Day}_{sin} = \sin\left(\frac{2\pi}{\text{t}_{day}} \times \text{t}\right) \tag{3.1}$$

$$\text{Day}_{cos} = \cos\left(\frac{2\pi}{\text{t}_{day}} \times \text{t}\right) \tag{3.2}$$

$$\text{Year}_{sin} = \sin\left(\frac{2\pi}{\text{t}_{year}} \times \text{t}\right) \tag{3.3}$$

$$\text{Year}_{cos} = \cos\left(\frac{2\pi}{\text{t}_{year}} \times \text{t}\right) \tag{3.4}$$

t_{day}: This represents the period of the day cycle. It determines the length of one complete cycle of the day.

t_{year}: This represents the period of the yearly cycle. It determines the length of one complete cycle of the year.

t: This variable represents the time input value.

Year_{sin}: This variable represents the sine component of the cyclical encoding for the year. It captures the variation of the yearly cycle over time.

Year_{cos}: This variable represents the cosine component of the cyclical encoding for the year. It captures the phase of the yearly cycle at a given time.

Day_{sin}: This variable represents the sine component of the cyclical encoding for the day. It captures the variation of the day cycle over time.

Day_{cos}: This variable represents the cosine component of the cyclical encoding for the day. It captures the phase of the day cycle at a given time.

While the cyclical encoding method proves beneficial, it does have its limitations. One inconvenience arises from converting a single piece of information into two features. Mathematically speaking, this can result in the algorithm assigning disproportionate weight to it. Decision tree-based algorithms, such as Random Forest, Gradient Boosted Trees, and XGBoost, construct their splitting rules based on individual features. Consequently, they may struggle to effectively process these two features simultaneously, despite the intended use of cosine and sine values as a unified coordinate system.[62]

3.2.4 Data scaling

When dealing with input variables that exhibit varying orders of magnitude, it becomes necessary to rescale them to ensure an effective learning process. The objective of rescaling is to preserve the original distribution of the variable in consideration while aligning its magnitude with that of the other variables. Two suitable options in such cases are min-max scaling and interquantile scaling[35].

The min-max scaling equation is given by:

$$x_{\text{minmax}} = \frac{x - \text{min}(x)}{\text{max}(x) - \text{min}(x)} \tag{3.5}$$

Similarly, the interquantile scaling equation is expressed as:

$$x_{\text{Q1Q3}} = \frac{x - Q1(x)}{Q3(x) - Q1(x)} \tag{3.6}$$

Here, $\text{min}(x)$, $\text{max}(x)$, $Q1(x)$, and $Q3(x)$ represent the minimum, maximum, 1st quartile, and 3rd quartile of the x variable, respectively.

Alternatively, if the variable x is assumed to follow a normal distribution, the z-score rescaling can be used to standardize the variable with zero mean and unit variance.

The z-score rescaling equation is given by:

$$x_z = \frac{x - \mu_x}{\sigma_x} \tag{3.7}$$

Here, μ_x and σ_x represent the mean and standard deviation of the x variable, respectively[35].

In our study on PV power forecasting, we utilized the widely adopted min-max scaler from the sklearn library, based on its extensive usage in the literature.

3.3 Feature Selection

Feature selection is essential in time series prediction, particularly in academia. It involves identifying the most informative features from a dataset, improving accuracy and efficiency. By selecting relevant features that capture underlying patterns, noise and irrelevant information are eliminated, reducing complexity and enhancing the model's generalization capability.

In our study, we will perform feature selection for multivariate models only to predict global irradiance, extracting insights and achieving more accurate predictions in time series analysis.

3.3.1 Feature Selection for The multivariant forecasting models

After performing data quality verifications on the acquired datasets, correlation analysis was conducted to evaluate the importance of input features on the output variable. This analysis involved calculating correlation coefficients between each feature and the target variable to measure the strength and direction of the linear relationship between them. Various metrics can be used to measure correlation, such as Pearson r correlation, Kendall rank correlation, Spearman rank correlation, and others. Among these, Pearson's r correlation is widely used for linearly related variables and can be calculated using the following equation:

$$r = \frac{n\sum(xy) - \sum x \sum y}{\sqrt{[n\sum(x^2) - (\sum x)^2][n\sum(y^2) - (\sum y)^2]}} \tag{3.8}$$

x and y represent the two variables being compared. $\sum$ denotes summation over all the data points. n represents the total number of values for the variables. r represents the Pearson's r correlation.

The equation calculates the covariance between the variables and divides it by the product of their standard deviations to quantify the strength and direction of the linear relationship between variables, providing a value between -1 and 1.[63]

To perform correlation-based feature selection, the correlation coefficients were calculated between each two features and resulted in the following correlation matrices:

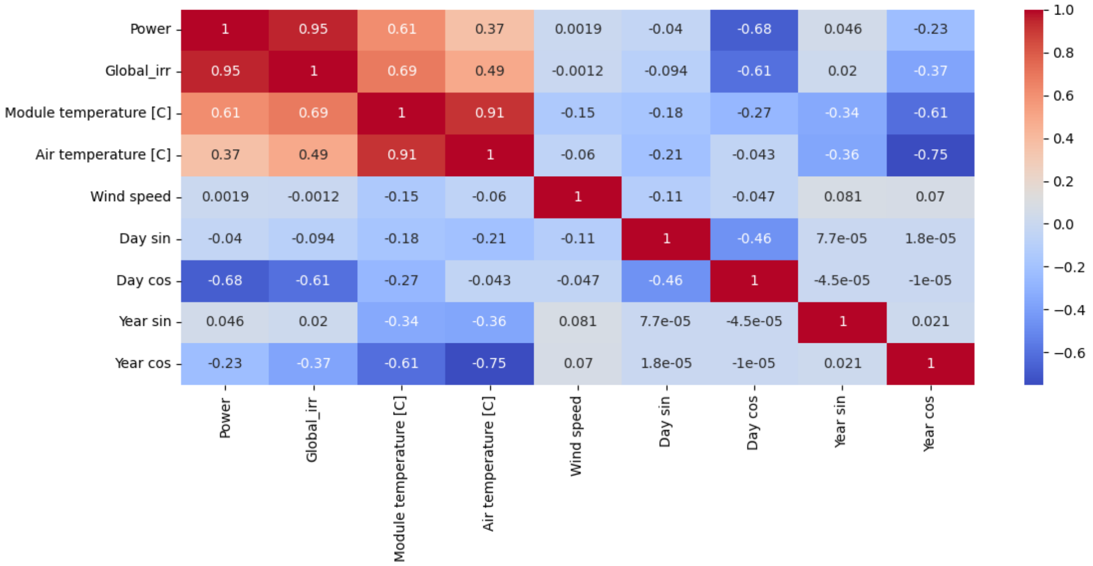

Figure 33. *Dhaya PV plant data correlation matrix.*

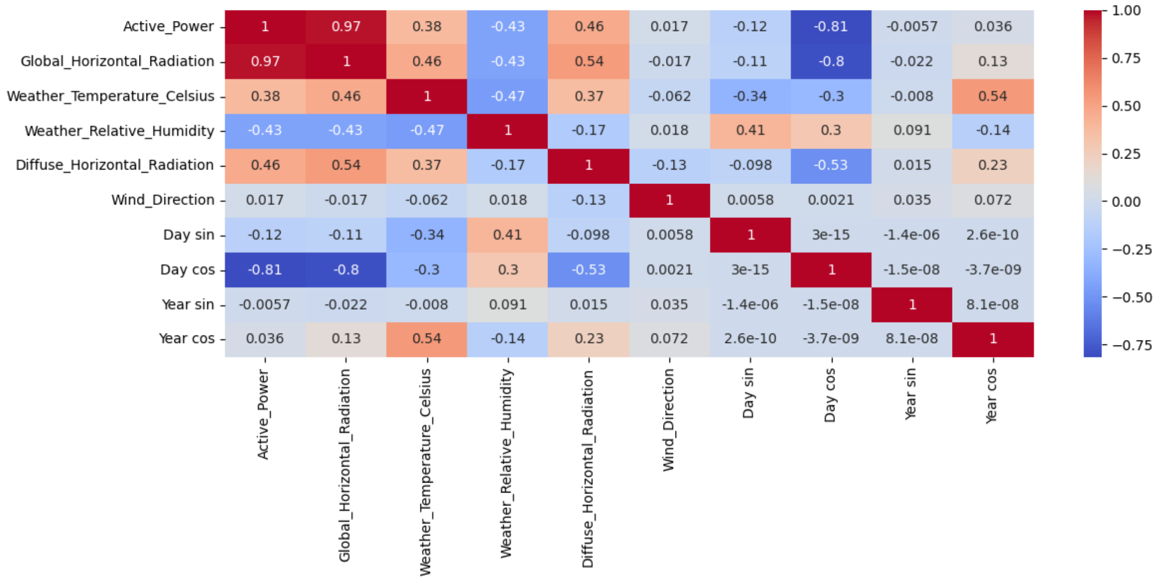

Figure 34. *DKASC data correlation matrix.*

After analyzing the input features and studying the correlation, insignificant parameters were removed from the input datasets. This step ensured that only the most relevant features (the ones with the highest absolute correlation values) were considered for further analysis and modeling.

The final set of features, which were considered significant, are:

1. **Module temperature**: This feature represents the temperature of the solar panels or photovoltaic (PV) modules. It is correlated to irradiance because as the module temperature increases, the efficiency of the PV cells tends to decrease. Higher temperatures can cause thermal losses and reduce the overall output power of the PV system, affecting the amount of energy generated from the available sunlight.
2. **Air \ weather temperature**: The air temperature refers to the temperature of the surrounding atmosphere. It is correlated to irradiance because temperature affects the air density, which in turn affects the transmission of sunlight. Higher temperatures generally lead to lower air densities, resulting in less scattering and absorption of sunlight by the atmosphere, and therefore, higher levels of irradiance.
3. **Day sin/cos (time in a day)**: This feature represents the time of day as a sine and cosine function. It captures the diurnal (daily) pattern of solar irradiance, as sunlight intensity varies throughout the day. The sinusoidal representation allows the model to account for the periodicity of daylight hours, which affects the available sunlight for power generation.
4. **Year sin/cos (time in a year)**: This feature encodes the time of year as a sine and cosine function. It captures the seasonal variations in irradiance due to changes in the Earth's tilt and its position relative to the sun. The angle of incidence of sunlight and the length of daylight hours vary with the time of year, influencing the amount of solar energy received.
5. **Weather relative humidity(for DKASC Data)**: Relative humidity is a measure of the moisture content in the air relative to its maximum capacity at a given temperature. It is correlated to irradiance because humidity affects the scattering and absorption of sunlight by water vapor and other particles in the atmosphere. Higher humidity levels can lead to increased cloud cover, which reduces the amount of direct sunlight reaching the solar panels, thus impacting irradiance.

global irradiance vs power generated

In order to investigate the relationship between generated power and irradiance in photovoltaic (PV) systems, a thorough analysis was conducted using the collected dataset. To visualize this relationship, a graph of irradiance versus power generated was plotted, as shown in figure 35.

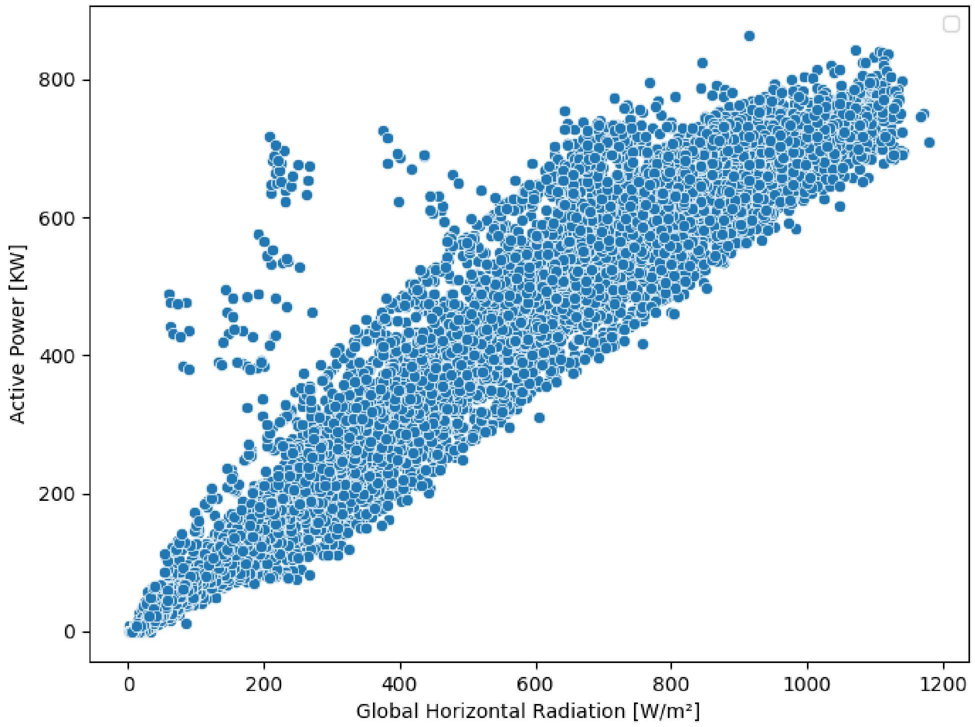

Figure 35. *global irradiance vs power generated in Dhaya PV plant data.*

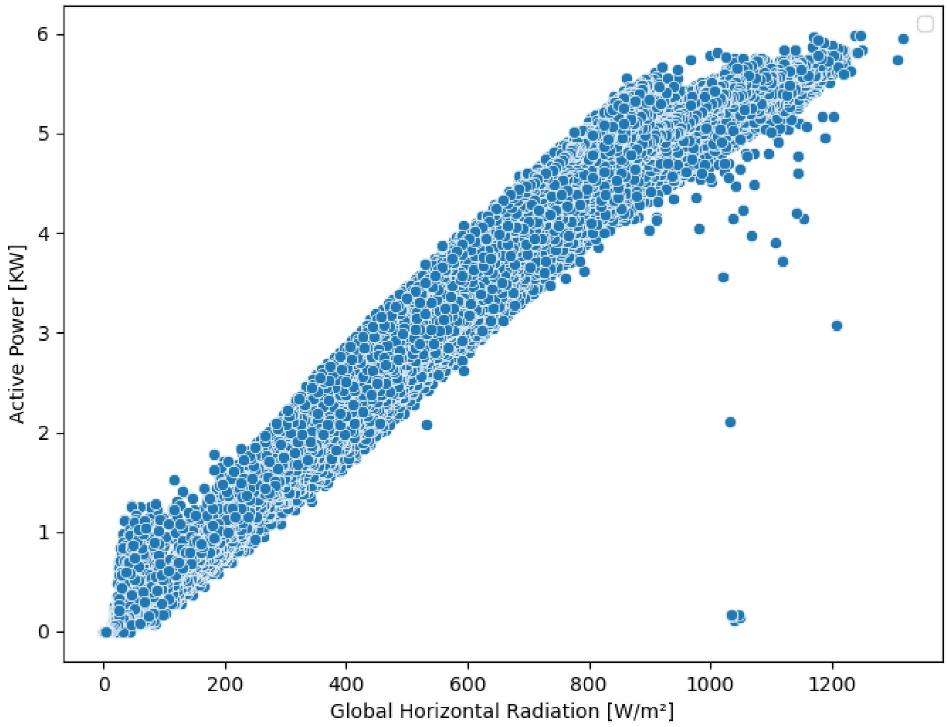

Figure 36. *global irradiance vs power generated in DKASC data.*

Upon analyzing the graph, it became evident that there is a strong linear relationship between irradiance and power generated. Building upon this observation, an interesting avenue for further exploration involves predicting irradiance levels. By utilizing machine learning techniques, specifically regression models, a predictive model can be developed to estimate the generated power levels. Once the irradiance values are predicted, a linear model can be applied to convert them into estimated power generated.

3.4 Conclusion

In conclusion, this chapter on feature engineering in PV power forecasting using machine learning emphasized the importance of data preparation and manipulation techniques. We discussed data presentation, handling missing values, addressing outliers, cycling coding for time, data scaling, and feature selection using correlation analysis. By employing these techniques, we ensured the quality and reliability of our forecasting models. Notably, our exploration of the relationship between global irradiance and generated power revealed the potential for predicting irradiance and converting it to power using a linear model. These findings contribute to the advancement of accurate and efficient PV power forecasting models, ultimately enabling better utilization of solar energy resources.

4. Results and Discussion

After thorough examination of the theoretical aspects of machine learning models used and metaheuristic algorithms for the tuning process, as well as conducting feature engineering to enhance the data and make it suitable for the learning process, we can now proceed to evaluate different machine learning models for the task of forecasting photovoltaic (PV) output power. Subsequently, we will compare the performance of these models and engage in a comprehensive discussion of the results.

As previously mentioned, we will employ an indirect approach to PV power forecasting. This approach involves forecasting the irradiance and subsequently developing a linear model to convert the forecasted irradiance into power.

4.1 Implementation Framework and Hyperparameter Fine-tuning

For this research study, we employed the Python programming language due to its simplicity and the availability of rich data analysis and machine learning libraries. Specifically, we utilized NumPy, Pandas, and Matplotlib for data manipulation and visualization. The Scikit-learn library provided tools for decision trees, SVR, and Random Forest models. XGBoost was employed for boosting algorithms, while TensorFlow's Keras interface facilitated the implementation of neural networks. These libraries were chosen for their versatility and extensive support for model training, evaluation, and hyperparameter optimization.

For hyperparameter tuning, we utilized the Differential Evolution (DE) algorithm from the Scipy library due to its robustness and customizable implementation. Additionally, the PySwarm library was used to leverage the Particle Swarm Optimization (PSO) algorithm. These algorithms allowed us to effectively explore and optimize the hyperparameter configurations of the machine learning models.

Owing to the constraints imposed by time limitations and insufficient computational power, a comprehensive tuning of hyperparameters was not performed across all models. Rather, a judicious selection was made, focusing solely on hyperparameters known to have a significant impact on the models. Furthermore, appropriate parameter boundaries were determined based on existing literature and the outcomes of preliminary experiments. Notably, the XGBoost and Random Forest models were excluded from the tuning process due to their extensive training time requirements, which would have posed a significant computational burden.

We conducted hyperparameter tuning on the following parameters to optimize the performance of our models.

Model	Hyperparameters
Decision Trees	- Historical Data Window Length - Max Depth - Min Sample Leaf - Min Sample Split
SVR	- Historical Data Window Length - C - Epsilon
LSTM	- Historical Data Window Length - Number of LSTM Units - Number of Neurons in Hidden Layer
BiLSTM	- Historical Data Window Length - Number of LSTM Units - Number of Neurons in Hidden Layer
GRU	- Historical Data Window Length - Number of GRU Units - Number of Neurons in Hidden Layer
CNN-LSTM	- Historical Data Window Length - Number of LSTM Units - Number of Neurons in Hidden Layer - Number of Convolutional Filters
Echo State	- Historical Data Window Length - Number of Echo State Units - Spectral Radius - Leaky - Connectivity - Number of Neurons in Hidden Laye

Table 5. Summary of Tuned Hyperparameters

To expedite the hyperparameter tuning process and minimize computational complexity, we opted to reduce the training data to 20% of its original size. This reduction allowed for shorter training times and decreased overall computational demands, particularly for models that required significant processing resources.

4.2 Single step Forecasting

4.2.1 On DKASC Dataset

The data was collected at 5-minute intervals, and our models were trained to predict the next 5-minute time step. Following the training and subsequent testing phases on the DKASC dataset, our model exhibited the following performance metrics on The Global irradiance forecasting .

Model	RMSE	MAE	MeAE	R2
Linear Regression	53.80	20.78	5.64	0.98
Linear SVR	56.13	19.48	5.56	0.97
Decision Trees	54.54	20.19	2.24	0.98
Random Forests	53.64	19.36	2.15	0.98
XGBoost	51.31	16.65	0.88	0.98
LSTM	52.71	19.17	3.14	0.98
Bi LSTM	52.27	19.04	2.45	0.98
GRU	51.44	17.65	2.31	0.98
CNN-LSTM	52.23	19.13	2.18	0.98
Echo-state Network	53.34	19.46	4.06	0.98

Table 6. Model Performance Metrics For Single-step Forecasting for DKASC dataset

The analysis of the provided table reveals several noteworthy observations regarding the performance of different models:

- XGBoost demonstrated superior performance across all metrics when compared to other models in the study.
- LSTM models exhibited competitive performance similar to that of XGBoost.
- GRU models showcased the best performance among all the LSTM models considered.
- Echo state networks performed closely to LSTM variants, despite having a significantly lower number of trainable parameters.
- Random Forests and decision trees outperformed deep learning methods in terms of the Median Absolute Error metric.
- Linear models, such as Linear Regression and LinearSVR, exhibited relatively lower performance as anticipated. However, their performance was still remarkably close to that of the other models.

These findings emphasize the effectiveness of XGBoost, LSTM, and Echo state models in the context of the analyzed metrics. Figures 37 to 46 further illustrate the obtained results.

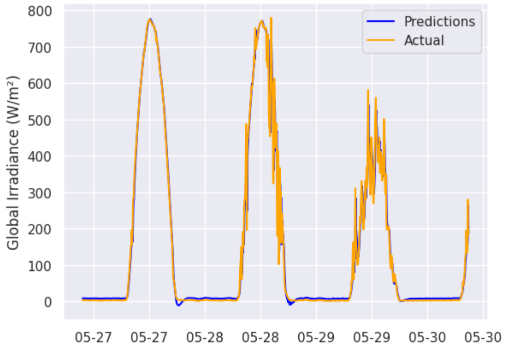

Figure 37. *Linear regression Predictions on DKASC dataset*

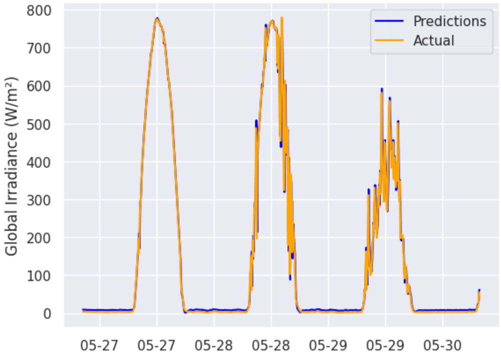

Figure 38. *LinearSVR Predictions on DKASC dataset*

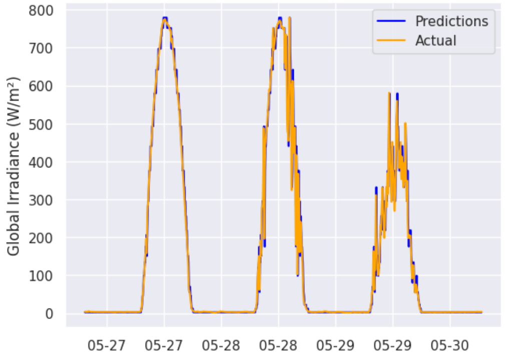

Figure 39. *Decision Trees Predtions on DKASC dataset*

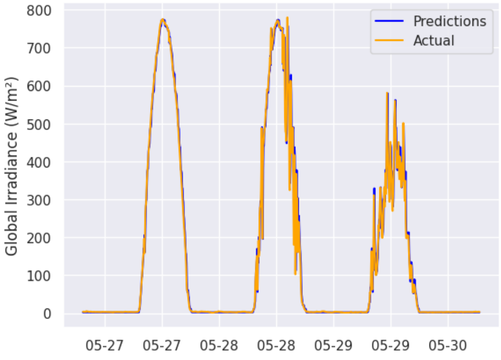

Figure 40. *Random Forests Predictions on DKASC dataset*

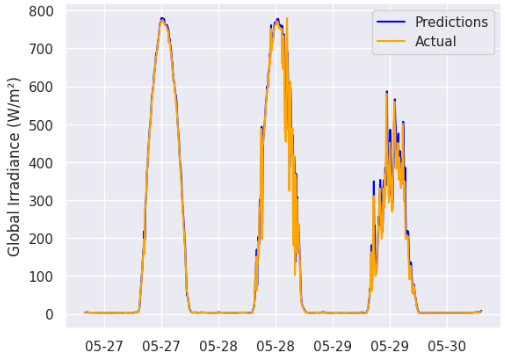

Figure 41. *XGBoost Predictions on DKASC datase*

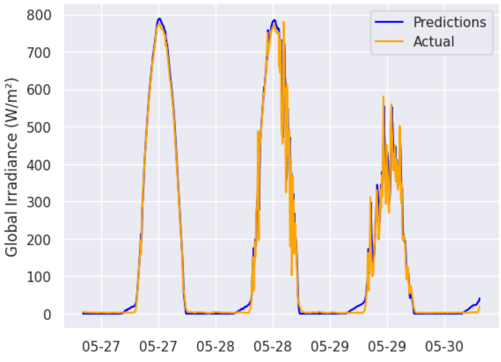

Figure 42. *Echo-state Network Predictions on DKASC dataset*

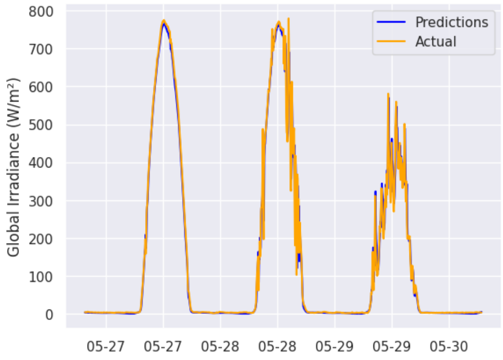

Figure 43. *LSTM Predictions on DKASC dataset*

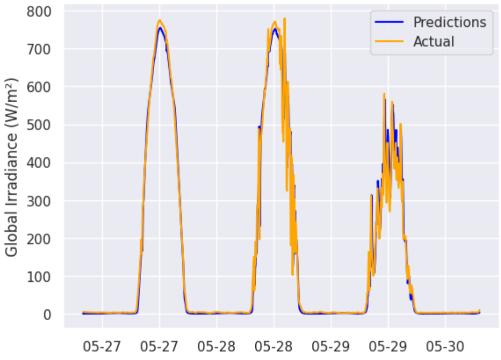

Figure 44. *BiLSTM Predictions on DKASC dataset*

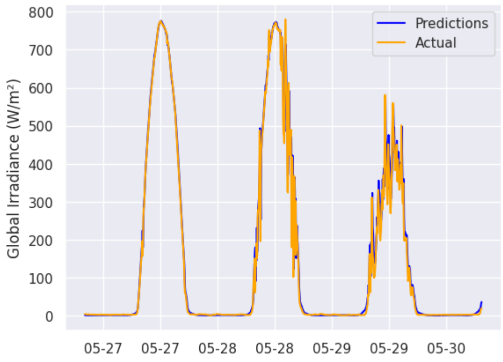

Figure 45. *GRU Predictions on DKASC dataset*

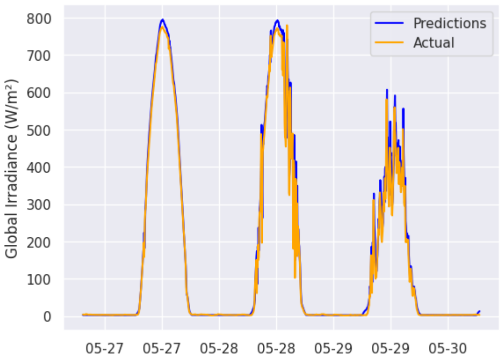

Figure 46. *CNN-LSTM Predictions on DKASC dataset*

4.2.2 On Dhya Dataset

The data was collected at 15-minute intervals, and our models were trained to predict the next 15-minute time step. Following the training and subsequent testing phases on the dataset, our model exhibited the following performance metrics on The Global irradiance forecasting .

Model	RMSE	MAE	MeAE	R2
Linear Regression	89.7	53.8	27.93	0.92
Linear SVR	90.30	53.41	26.86	0.92
Decision Trees	96.22	59.22	32.52	0.911
Random Forests	93.31	56.14	29.08	0.91
XGBoost	88.92	51.49	21.98	0.92
LSTM	88.53	51.13	22.05	0.92
Bi LSTM	93.10	61.08	37.28	0.91
GRU	88.48	51.47	23.27	0.92
CNN-LSTM	95.36	63.27	39.44	0.91
Echo-state Network	90.86	55.71	29.89	0.92

Table 7. Model Performance Metrics For Single-step Forecasting for Dhaya dataset

The analysis of the provided table reveals several noteworthy remarks regarding the performance of different models:

- XGBoost and LSTM models continued to perform the best.
- Certain LSTM models such as GRU and vanilla LSTM outperformed XGBoost slightly on the RMSE score.
- CNN-LSTM yielded the lowest performance among the LSTM models examined.
- Echo state networks performed similarly to XGBoost and various LSTM variants.
- Linear models performed well, closely matching the performance of deep learning models.
- Linear models outperformed both Random Forests and decision trees across all metrics.

The error observed in this study was higher compared to forecasting 5 minutes ahead.The higher error can be attributed to the smaller resolution, which may cause error accumulation.

Figures 47 to 56 further illustrate the obtained results.

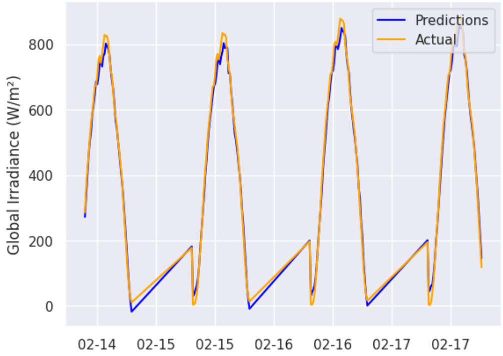

Figure 47. *Linear regression Predictions on Dhaya dataset*

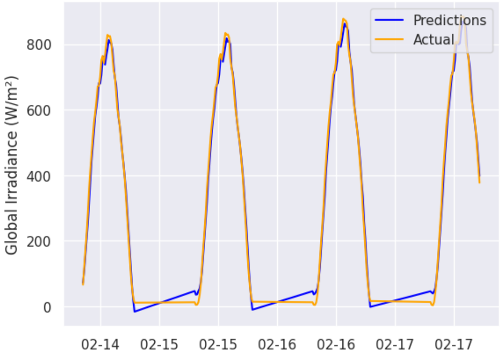

Figure 48. *LinearSVR Predictions on Dhaya dataset*

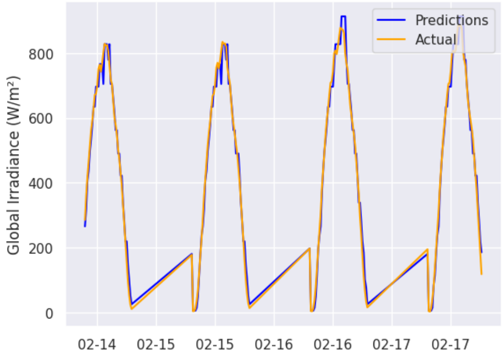

Figure 49. *Decision Trees Predictions on Dhaya dataset*

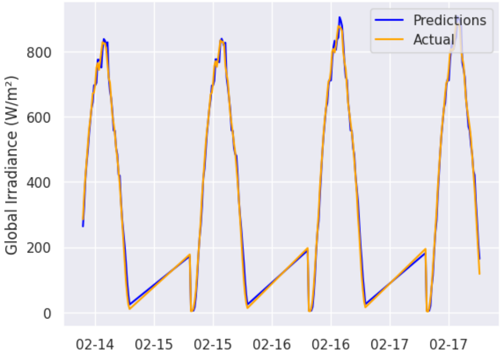

Figure 50. *Random Forests Predictions on Dhaya dataset*

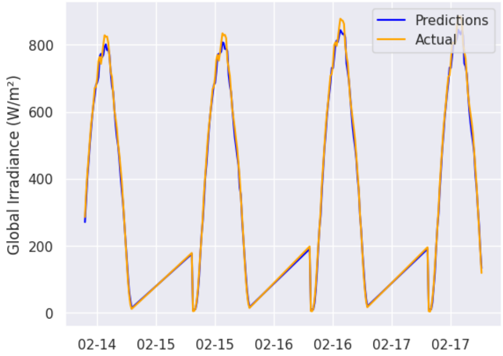

Figure 51. *XGBoost Predictions on Dhaya dataset*

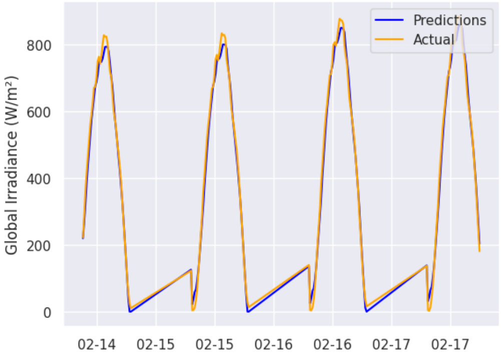

Figure 52. *Echo-state Network Predictions on Dhaya dataset*

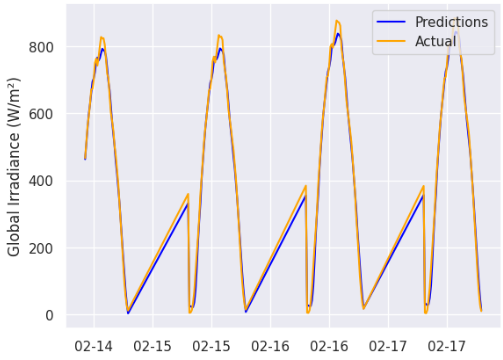

Figure 53. *LSTM Predictions on Dhaya dataset*

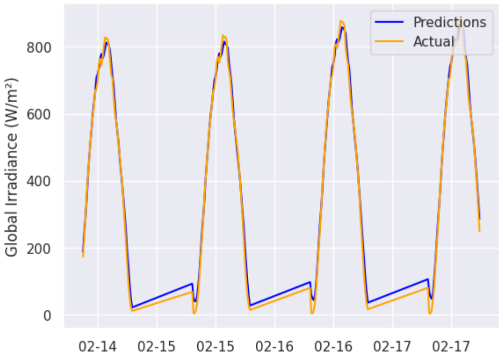

Figure 54. *BiLSTM Predictions on Dhaya dataset*

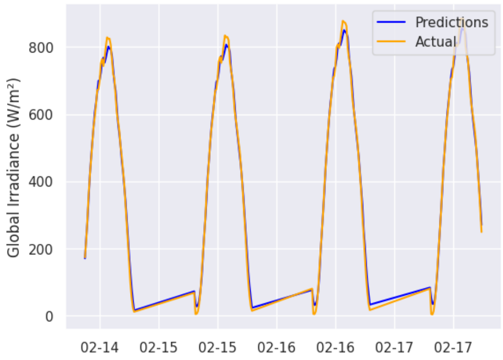

Figure 55. *GRU Predictions on Dhaya dataset*

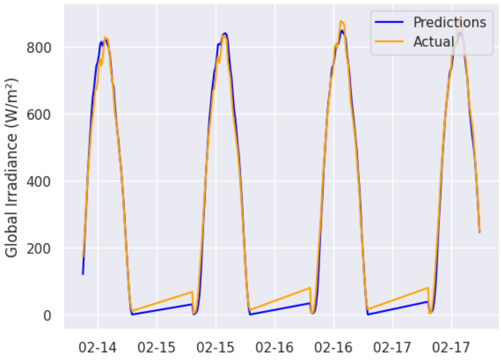

Figure 56. *CNN-LSTM Predictions on Dhaya dataset*

4.3 Multiple steps Forecasting

4.3.1 On DKASC Dataset

This study utilized a 15-minute resampling interval instead of 5 minutes. The objective was to predict an 8-step, two-hour ahead period. Table 8 presents the results for the employed models.

Model	RMSE	MAE	MeAE	R2
Linear Regression	119.99	74.24	42.84	0.88
Linear SVR	128.95	70.40	32.19	0.86
Decision Trees	102.48	47.30	3.92	0.91
Random Forests	98.24	44.38	3.48	0.92
XGBoost	91.61	37.84	3.00	0.93
LSTM	93.04	40.59	3.96	0.93
Bi LSTM	95.50	42.98	4.86	0.93
GRU	93.15	41.42	4.47	0.93
CNN-LSTM	94.70	41.32	4.19	0.93
Echo-state Network	101.94	48.86	11.23	0.91

Table 8. Model Performance Metrics For Multi-step Forecasting for DKASC dataset

As the prediction horizon expands, the prediction error increases consistently, aligning with our expectations. We remark the following:

1. XGboost consistently outperforms all other models in terms of all evaluated metrics.
2. LSTM models show comparable performance, following closely behind Xgboost.
3. Random forests, Decision trees, and Echo state models rank lower in predictive accuracy, displaying a similar pattern to one-step ahead prediction.
4. The performance of the Linear models deteriorates, resulting in higher prediction errors.

This highlights the limitations of linear models in making long-term predictions, emphasizing the non-linear relationship among past observations.

Figures 57 to 66 illustrate the difference between the predicted irradiance and the real value for the last 2 hours.

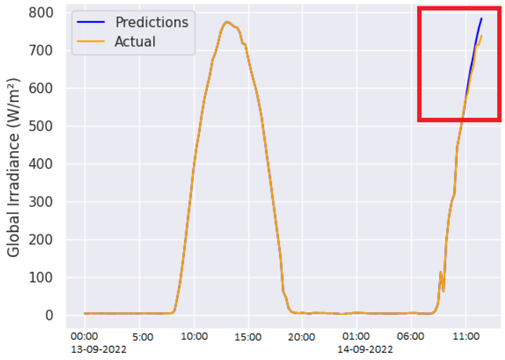

Figure 57. *Linear regression Predictions on DKASC dataset*

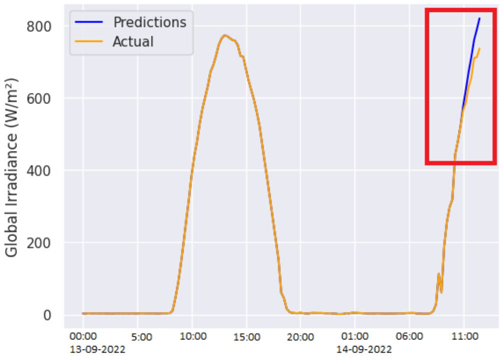

Figure 58. *LinearSVR Predictions on DKASC dataset*

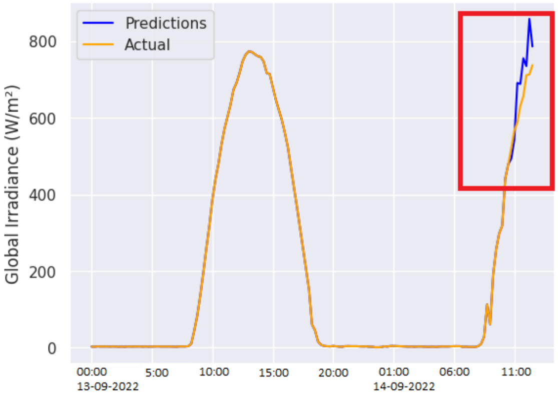

Figure 59. *Decision Trees Predictions on DKASC dataset*

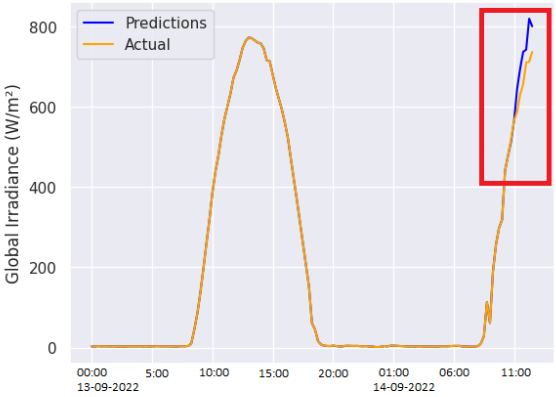

Figure 60. *Random Forests Predictions on DKASC dataset*

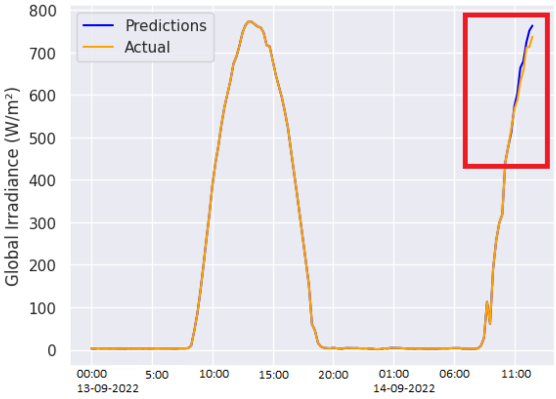

Figure 61. *XGBoost Predictions on DKASC dataset*

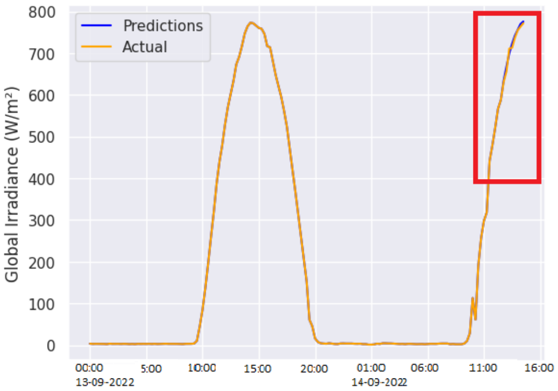

Figure 62. *LSTM Predictions on DKASC dataset*

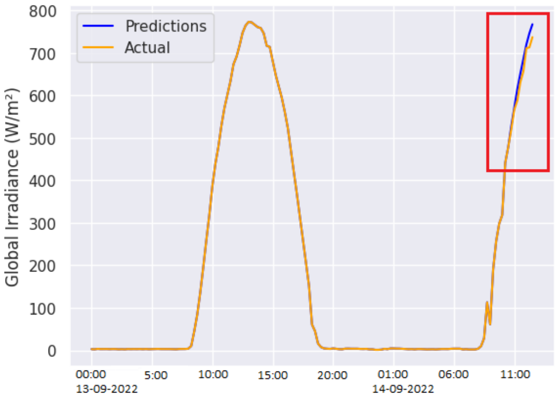

Figure 63. *BiLSTM Predictions on DKASC dataset*

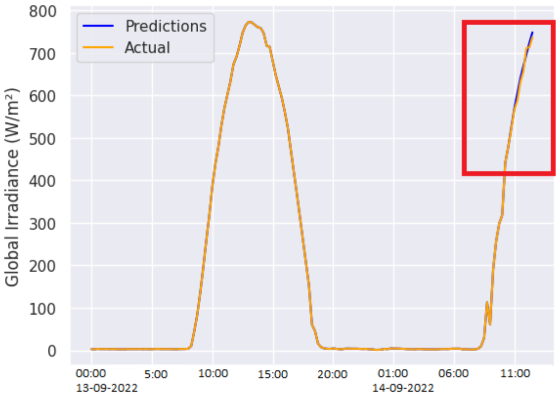

Figure 64. *GRU Predictions on DKASC dataset*

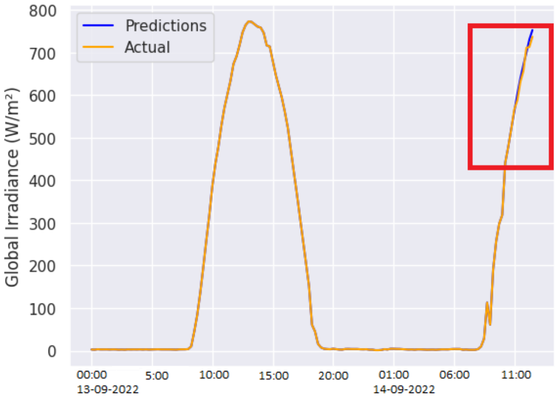

Figure 65. *CNN-LSTM Predictions on DKASC dataset*

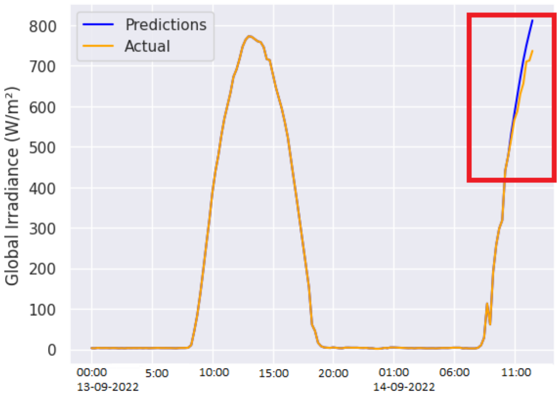

Figure 66. *Echo-state Network Predictions on DKASC dataset*

4.3.2 On Dhya Dataset

The Algerian dataset was collected at regular intervals of 15 minutes. We maintained the dataset in its original form and utilized it to forecast the subsequent 2-hour period, which corresponds to 8 future time steps. Our objective was to compare the outcomes obtained from models trained on the DKASC dataset with those generated using the Algerian dataset. The resultant findings are presented in table 9.

Model	RMSE	MAE	MeAE	R2
Linear Regression	157.99	108.14	70.60	0.74
Linear SVR	163.84	106.47	61.56	0.73
Decision Trees	185.79	127.55	83.83	0.65
Random Forests	177.48	122.71	83.25	0.68
XGBoost	147.89	95.55	55.03	0.78
LSTM	147.03	99.01	62.59	0.78
Bi LSTM	148.04	98.56	60.38	0.78
GRU	149.98	99.30	57.75	0.77
CNN-LSTM	151.46	109.46	80.57	0.77
Echo-state Network	152.27	101.18	63.45	0.76

Table 9. Model Performance Metrics For Multi-step Forecasting for Dhaya dataset

The analysis of the models trained on the DKASC dataset reveals a significant increase in error compared to the previous models. Despite being sampled at the same interval and predicting the same horizon, the observations can be organized into the following list:

- LSTM models and XGBoost demonstrate the best performance, closely competing with each other on the RMSE score.
- XGBoost outperforms other models on MAE (Mean Absolute Error) and MeAE (Median Absolute Error).
- Echo state networks follow the LSTM models and XGBoost, showing relatively good performance.
- Linear models such as SVR (Support Vector Regression) and linear regression exhibit a considerable deviation from the other models, displaying a large gap on all evaluation metrics.
- Decision trees and random forests perform poorly on this dataset compared to the previous one, indicating a decline in their predictive capabilities.

Figures 67 to 76 illustrate the difference between the predicted irradiance and the real value for the last 2 hours.

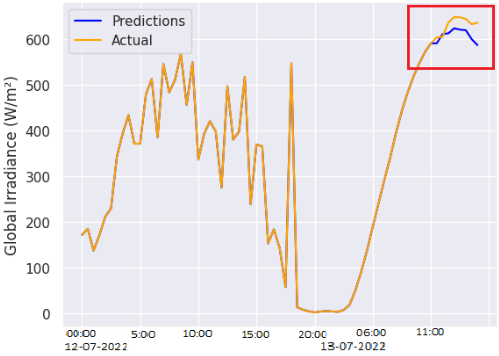

Figure 67. *Linear regression Predictions on Dhaya dataset*

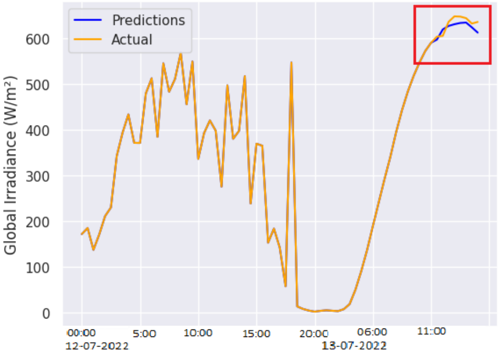

Figure 68. *LinearSVR Predictions Fon Dhaya dataset*

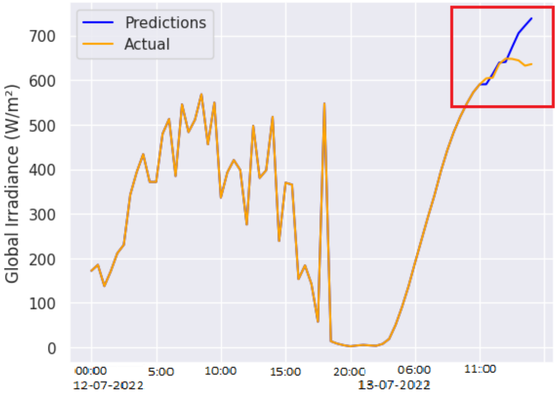

Figure 69. *Decision Trees Predictions on Dhaya dataset*

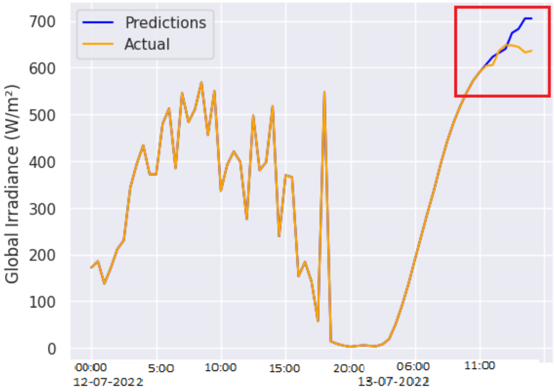

Figure 70. *Random Forests Predictions on Dhaya dataset*

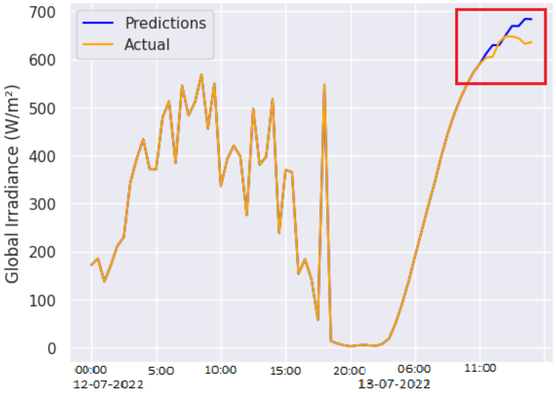

Figure 71. *XGBoost Predictions on Dhaya dataset*

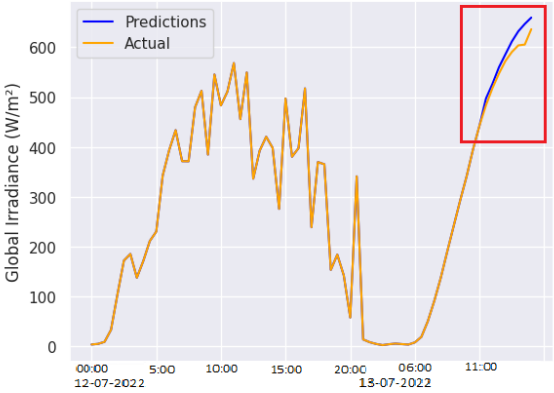

Figure 72. *LSTM Predictions on Dhaya dataset*

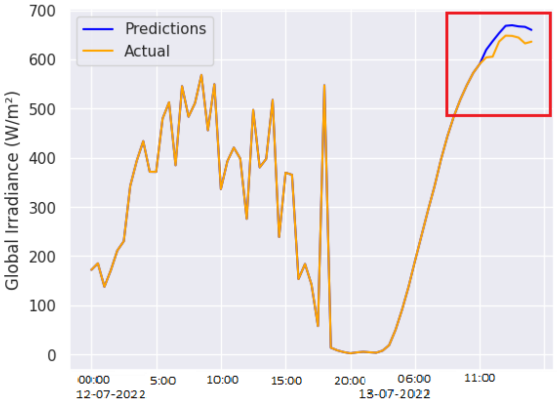

Figure 73. *BiLSTM Predictions on Dhaya dataset*

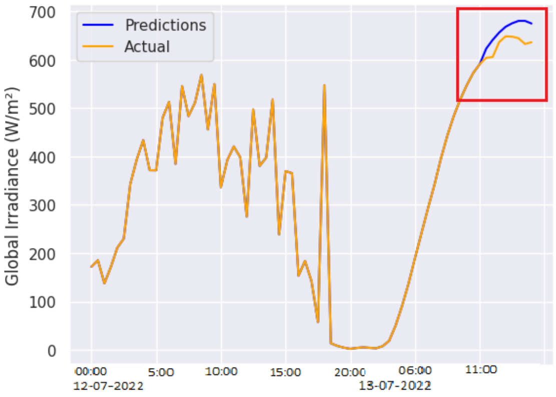

Figure 74. *GRU Predictions on Dhaya dataset*

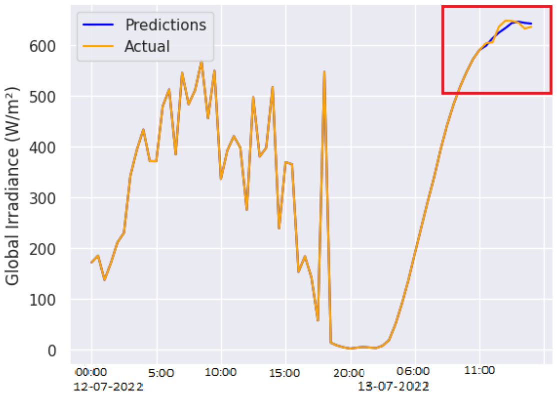

Figure 75. *CNN-LSTM Predictions on Dhaya dataset*

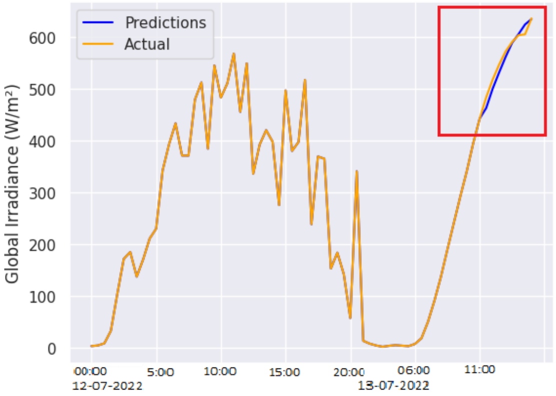

Figure 76. *Echo-state Network Predictions on Dhaya dataset*

4.4 From Irradiance To Power

To obtain the output power predictions from the irradiance, a simple linear regression model was employed. This choice was justified by the observed linear relationship between the irradiance and power, as discussed in Chapter 3.

4.4.1 DKASC Dataset

For the DKASC dataset, the following equation was derived:

$$Power = 0.00514 \times Irradiance + 0.0417 \tag{4.1}$$

The model performed as follows on the testing data:

Metric	**Value**
Root Mean Squared Error	0.380
Mean Absolute Error	0.18
Median Absolute Error	0.055
R-squared Score	0.96

Table 10. Irradiance to Power model Performance on DKASC dataset

4.4.2 Dhaya Dataset

For the Dhaya dataset, the linear regression yielded the following equation:

$$Power = 0.7951 \times Irradiance + 28.083 \tag{4.2}$$

The model performed as follows on the testing data:

Metric	**Value**
Root Mean Squared Error	68.51
Mean Absolute Error	54.73
Median Absolute Error	42.13
R-squared Score	0.93

Table 11. Irradiance to Power model Performance on Dhaya dataset

The discrepancy in error rates observed between the linear regression models applied to the DKASC and Dhaya datasets can be ascribed to the contrasting power capabilities of the Sanyo Power Center (with a maximum output power of 6.3 kW) and the Dhaya Skid 1 (with a maximum output power of 880 kW). By computing the ratio of the error to the respective maximum power values, it is evident that the DKASC dataset exhibits an error ratio of 6%, while the Dhaya dataset demonstrates a slightly higher error ratio of 7%. Consequently, it can be inferred that the accuracy of the linear regression models is relatively similar.

Subsequently, after identifying the most optimal model for global irradiance forecasting, the aforementioned equations can be employed to predict power output. While the direct method of power forecasting may yield superior outcomes due to the potential error accumulation when forecasting irradiance prior to power, the indirect approach surpasses the direct approach in terms of its independence from the specific PV plant. By adopting the indirect approach, the flexibility to modify the PV plant configuration or incorporate additional modules without necessitating the retraining of extensive models is achieved. Instead, a linear regression can be conducted to forecast power, providing a practical and efficient alternative in power prediction tasks.

4.5 Comparative Analysis of Differential Evolution and Particle Swarm Optimization

As previously indicated, a reduction of 20% to 40% in the training data was necessary for certain models due to their high time complexity during training.

When employing a large population size, both Differential Evolution (DE) and Particle Swarm Optimization (PSO) converged to hyperparameters that yielded similar performance, with DE occasionally exhibiting slightly superior performance, while PSO performed better in other instances.Multiple runs of the tuning procedure were conducted to ensure the stability of the solution.

However, it is important to note the following key results and important remarks:

1. DE consistently terminated with fewer iterations and function evaluations, indicating its efficiency compared to PSO.
2. In each Algorithm run, the results varied, highlighting the non-convex nature of the optimization problem with multiple local minima.
3. The performance of the hyperparameters remained consistent across Algorithm runs,

reinforcing the stability and the robustness of the algorithms and their capacity to deliver solutions that are in close proximity to the global optimum.

4. The discrepancies in results were attributed to the stochastic nature of the algorithms, as they initiated from different locations within the search space during each run.

In light of these findings, the best-performing hyperparameters were selected and employed in the training of the aforementioned models.

Additionally, it should be emphasized that when we reduced the population size for both DE and PSO:

- DE continued to generate hyperparameters with comparable performance to those obtained with a larger population size, indicating its robustness.
- PSO, on the other hand, started to exhibit premature convergence, terminating before reaching the same objective function value attained by DE. This suggests that a larger population size is more critical for PSO to achieve optimal results.

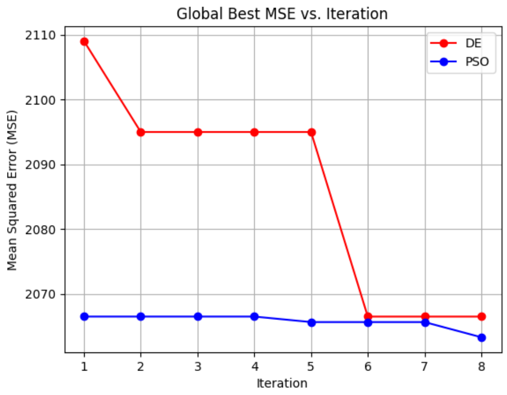

Figure 77. *PSO vs DE on Decision Trees P=40*

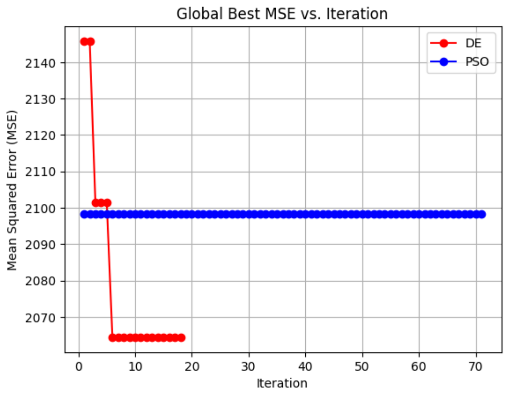

Figure 78. *PSO vs DE on Decision Trees P=10*

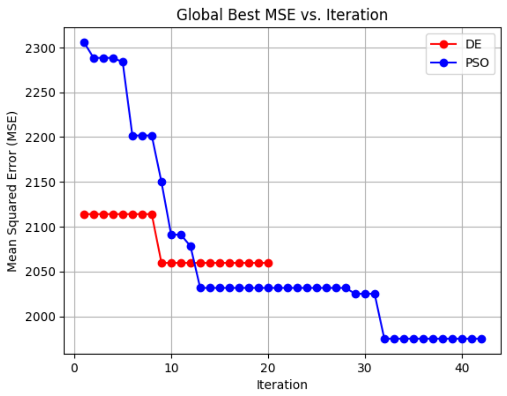

Figure 79. *PSO vs DE on LinearSVR P=30*

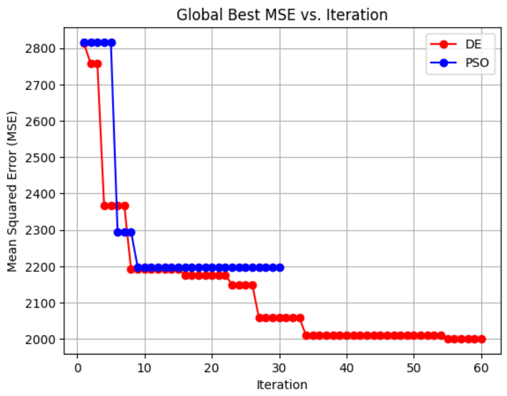

Figure 80. *PSO vs DE on LinearSVR P=9*

It is noteworthy to observe that the disparity in the outcomes obtained and the computational iterations performed was relatively small. Both algorithms exhibited commendable performance within a reasonable timeframe, surpassing the capabilities of grid search, which necessitates significantly larger computational iterations. Furthermore, random search also may be outperformed by the aforementioned algorithms since it may require additional iterations due to its lack of knowledge transfer between iterations.

4.6 Discussion and Conclusion

In our study, we compared several machine learning models for global irradiance forecasting on different datasets and prediction horizons. Our results consistently showed that XGBoost performed the best among all the models.

When it comes to LSTM variants, we observed similar performance across different datasets and prediction horizons. However, within this category, the GRU (Gated Recurrent Unit) model outperformed other LSTM models. GRU is a type of recurrent neural network that exhibits similar capabilities to LSTM but with a simpler architecture. It has fewer trainable parameters compared to LSTM, making it a lightweight neural network. This characteristic is beneficial as it reduces the computational complexity and memory requirements during training and inference.

Furthermore, we found that Echo State Networks (ESNs) demonstrated good performance despite not being superior to LSTM models. ESNs are a type of recurrent neural network where the recurrent connections are randomly initialized and remain fixed during training. They require far fewer trainable parameters, even fewer than GRU. This feature makes ESNs an attractive option for hyperparameter tuning, as they allow for exploration of a larger search space without extensive computational power requirements. Training ESNs is generally easier and faster compared to LSTM models due to their simplicity.

On the other hand, linear models such as Support Vector Regression (SVR) and linear regression showed promising results for short-term, single-step predictions. However, their performance significantly degraded for multi-step predictions, as the errors accumulated more rapidly over time. This limitation makes linear models less suitable for tasks requiring long-term forecasting accuracy.

We also evaluated decision trees and random forests, which performed well on the DKASC dataset but poorly on the Dhaya dataset. This discrepancy highlights their sensitivity to the quality and characteristics of the training data. Decision trees and random forests tend to be less robust and may not generalize well to different datasets. In contrast, the deep learning models, including LSTM variants and XGBoost, demonstrated more consistent performance across different datasets, indicating their ability to capture complex patterns and generalize effectively.

Regarding the employed metaheuristic search methods, it is worth noting that they exhibit comparable efficacy when applied with a substantial population size. Notably, Differential Evolution tends to terminate marginally earlier than Particle Swarm Optimization. Nevertheless, when the population size is decreased, PSO starts yielding inconsistent outcomes across multiple runs and becomes susceptible to premature convergence. This observation implies that Differential Evolution presents itself as a promising candidate for future investigations on population size reduction strategies, as it has demonstrated consistent performance under such circumstances.

In addition, it is noteworthy that the models trained on the DKASC dataset exhibit superior performance compared to those trained on the Algerian Dhaya dataset, despite employing similar variables for prediction, employing the same sampling time and prediction horizon. This discrepancy in performance can potentially be attributed to the presence of high noise levels within the Algerian dataset, coupled with a substantial number of missing values and Low sampling resolution. These factors likely introduce additional complexities and uncertainties, thereby impeding the effectiveness of the models trained on the Algerian Dhaya dataset.

Conclusion

This research aimed to predict PV power generation by utilizing various machine learning techniques fine-tuned with metaheuristic search methods. The study evaluated and compared the performance of different models, including Random Forest, XGBoost, Echo State Networks (ESNs), LSTM variants, and linear models, using 4 assessment metrics.

The models were trained on different datasets and used to predict different horizons. Specifically, we utilized the DKASC and Dhaya (Algeria) datasets for forecasting horizons of 5 minutes, 15 minutes, and 2 hours.

Consistently, the results demonstrated that XGBoost outperformed other models in forecasting global irradiance. Among the LSTM variants, the GRU model exhibited superior performance due to its simpler architecture and fewer trainable parameters. ESNs also performed well but were not as effective as LSTM models. ESNs are computationally efficient due to their lightweight architecture and fewer trainable parameters.

Linear models like SVR and linear regression showed good results for short-term, single-step predictions but had reduced accuracy for multi-step predictions, limiting their suitability for long-term forecasting.

The performance of decision trees and random forests varied depending on the dataset, indicating sensitivity to data quality and characteristics. On the other hand, deep learning models such as LSTM variants and XGBoost consistently performed well across different datasets, demonstrating their ability to capture complex patterns and generalize effectively.

Regarding metaheuristic search methods, Differential Evolution and Particle Swarm Optimization showed comparable effectiveness when used with a significant population size. However, reducing the population size caused PSO to produce inconsistent results and become prone to premature convergence. Differential Evolution emerged as a promising option for future investigations into population size reduction strategies due to its consistent performance under such circumstances.

It is worth noting that models trained on the DKASC dataset consistently outperformed those trained on the Algerian Dhaya dataset. This performance difference can be attributed to the presence of high noise levels, missing values, low sampling resolution, and outliers in the Algerian dataset, introducing additional complexities and uncertainties that hindered the models' effectiveness.

If the energy sector in Algeria shows a strong interest in implementing photovoltaic (PV) energy systems on a large scale, it is crucial to initiate the collection of reliable and accurate datasets using high-resolution sensors of superior quality. Additionally, incorporating a wider range of meteorological variables is vital to improve the precision of predicting irradiance, which in turn enables more accurate estimation of power production.

Overall, this study provides valuable insights into the performance and characteristics of various machine learning techniques for forecasting PV power generation. These findings can serve as a guide for future research and applications in the field of renewable energy forecasting.

Future Work

The following is a list of suggested future research directions:

1. Utilize population size reduction strategies in Differential Evolution (DE) for improved efficiency in hyperparameter tuning.
2. Incorporate Hybrid Echo State Networks (ESNs) as a promising approach for predicting chaotic time series.
3. Integrate DE as a training method for Recurrent Neural Network (RNN) models, such as Long Short-Term Memory (LSTM), to mitigate the vanishing or exploding gradient problem and enable longer temporal dependencies.
4. Explore the performance of alternative metaheuristic algorithms, including Grey Wolf Optimization, Bee Colony Optimization, and Genetic Algorithms, in hyperparameter tunning .

References

[1] Jimeng Shi, Mahek Jain, and Giri Narasimhan. "Time Series Forecasting (TSF) Using Various Deep Learning Models". In: (2022). arXiv: 2204.11115 [cs.LG]. URL: https://arxiv.org/abs/2204.11115.

[2] Qian Li, Zhou Wu, Rui Ling, Liang Feng and Kai Liu. "Multi-reservoir echo state computing for solar irradiance prediction: A fast yet efficient deep learning approach". In: *Applied Soft Computing* 95 (2020), p. 106481. ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2020.106481. URL: https://www.sciencedirect.com/science/article/pii/S1568494620304208.

[3] Rui Liu Lu Peng Shan Liu and Lin Wang. "Effective long short-term memory with differential evolution algorithm for electricity price prediction". In: *Energy* 162 (2018), pp. 1301–1314. ISSN: 0360-5442. DOI: https://doi.org/10.1016/j.energy.2018.05.052. URL: https://www.sciencedirect.com/science/article/pii/S0360544218308727.

[4] Matthew Sabas. "History of Solar Power". In: *Institute for Energy Research* (2016). Accessed: May 28, 2023. URL: https://www.instituteforenergyresearch.org/renewable/solar/history-of-solar-power/.

[5] Lewis Fraas. "Chapter 1: History of Solar Cell Development". In: *Low Cost Solar Electric Power*. 1st. Springer, June 2014. Chap. 1, pp. 1–12. DOI: 10.1007/978-3-319-07530-3_1. URL: https://www.researchgate.net/publication/274961489_Chapter_1_History_of_Solar_Cell_Development.

[6] Florida Solar Energy Center. *History of Photovoltaics*. Accessed: May 28, 2023. URL: https://www.fsec.ucf.edu/en/consumer/solar_electricity/basics/history_of_pv.htm.

[7] IEA. *Solar PV*. Accessed: May 13, 2023. IEA, Paris. September 2022. URL: https://www.iea.org/reports/solar-pv.

[8] Our World in Data. *Solar power generation*. Retrieved: 2022-07-08, Accessed: May 15, 2023. Data published by BP Statistical Review of World Energy; Ember. URL: https://ourworldindata.org/grapher/solar-energy-consumption?tab=chart&time=2005..latest&country=AUS~USA~OWID_WRL~DZA.

[9] R. Ahmed, V. Sreeram, Y. Mishra and M.D. Arif. “A review and evaluation of the state-of-the-art in PV solar power forecasting: Techniques and optimization”. In: *Renewable and Sustainable Energy Reviews* 124 (2020), p. 109792. ISSN: 1364-0321. DOI: 10.1016/j.rser.2020.109792. URL: https://doi.org/10.1016/j.rser.2020.109792.

[10] Our World in Data. *Annual CO emissions*. Retrieved: 2023-04-28, Accessed: May 15, 2023. Data publisher’s source Global Carbon Project. URL: https://ourworldindata.org/grapher/annual-co2-emissions-per-country?country=~OWID_WRL.

[11] F. Heineke, N. Janecke, H. Klärner, F. Kühn, H. Tai and R. Winter. *Renewable-energy development in a net-zero world*. Accessed: May 13, 2023. McKinsey & Company. October 28, 2022. URL: https://www.mckinsey.com/industries/electric-power-and-natural-gas/our-insights/renewable-energy-development-in-a-net-zero-world.

[12] Samy Khalil and Usama Rahoma. “Verification of Solar Energy Measurements by (ERA-5) and Its Impact on Electricity Costs in North Africa”. In: *International Journal of Astronomy and Astrophysics* 12 (Jan. 2022), pp. 301–327. DOI: 10.4236/ijaa.2022.124018.

[13] Solargis. *Photovoltaic Electricity Potential, Middle East and North Africa*. Downloaded from Solargis, Accessed: May 15, 2023. 2020. URL: https://solargis.com/maps-and-gis-data/download/middle-east-and-north-africa.

[14] M. Hall. “Solar dried up in North Africa during 2020”. In: *pv magazine* (May 2021). Accessed: May 13, 2023. URL: https://www.pv-magazine.com/2021/05/24/solar-dried-up-in-north-africa-during-2020/.

[15] Our World in Data. *Solar power generation, africa*. Retrieved: 2022-07-08, Accessed: May 15, 2023. Data published by BP Statistical Review of World Energy; Ember. URL: https://ourworldindata.org/grapher/solar-energy-consumption?tab=chart&time=2010..latest&country=~Africa+%5C%28BP%5C%29.

[16] M. Hochberg. *Algeria charts a path for renewable energy sector development*. Accessed: May 24, 2023. Middle East Institute. Oct. 2020. URL: https://www.mei.edu/publications/algeria-charts-path-renewable-energy-sector-development.

[17] M. Pennarts. *Renewable Energy in Algeria: Potential for Stability*. Accessed: May 25, 2023. The Borgen Project. Jan. 2023. URL: https://borgenproject.org/renewable-energy-in-algeria-potential-for-stability/.

[18] Solargis. *Photovoltaic Electricity Potential, Algeria*. Downloaded from Solargis, Accessed: May 25, 2023. 2020. URL: https://solargis.com/maps-and-gis-data/download/algeria.

[19] Our World in Data. *Solar power generation, Algeria*. Retrieved: 2022-07-08, Accessed: May 25, 2023. Data published by BP Statistical Review of World Energy; Ember. URL: https://ourworldindata.org/grapher/solar-energy-consumption?tab=chart&time=2010..latest&country=~DZA.

[20] *Algeria's location*. Google Maps. URL: https://goo.gl/maps/HUP59iueMN1NsxXx7.

[21] K. Mertens. *Photovoltaics Fundamentals, Technology and Practice*. Trans. by Gunther Roth. Germany: Münster University of Applied Sciences, 2014. Chap. 1, p. 13. URL: http://ndl.ethernet.edu.et/bitstream/123456789/87792/3/Photovoltaics%5C%20Fundamental%5C%20and.pdf.

[22] Energy Education. *Photovoltaic System*. Accessed: May 26, 2023. URL: https://energyeducation.ca/encyclopedia/Photovoltaic_system.

[23] E. A. Franklin. *Solar Photovoltaic (PV) System Components*. Extension Arizona. May 2018. URL: https://extension.arizona.edu/pubs/solar-photovoltaic-pv-system-components.

[24] A. Khaligh and O. C. Onar. "Energy Sources". In: *Power Electronics Handbook*. Ed. by M. H. Rashid. 3rd. Butterworth-Heinemann, 2011, pp. 1289–1330. ISBN: 9780123820365. URL: https://doi.org/10.1016/B978-0-12-382036-5.00045-8.

[25] Genus Innovation. *What Do You Need to Know About the Different Types of Solar PV Systems?* Accessed: May 26, 2023. URL: https://www.genusinnovation.com/what-do-you-need-to-know-about-the-different-types-of-solar-pv-systems.

[26] Energy Education. *Types of Photovoltaic Cells*. Accessed: May 26, 2023. URL: https://energyeducation.ca/encyclopedia/Types_of_photovoltaic_cells.

[27] Nikolay Chuchvaga , Kairat Zholdybayev , Kazybek Aimaganbetov , Sultan Zhantuarov and Abay Serikkanov. "Development of Hetero-Junction Silicon Solar Cells with Intrinsic Thin Layer: A Review". In: *Coatings* 13.4 (2023). ISSN: 2079-6412. DOI: 10.3390/coatings13040796. URL: https://www.mdpi.com/2079-6412/13/4/796.

[28] Thi Ngoc Nguyen and Felix Müsgens. "What drives the accuracy of PV output forecasts?" In: *Applied Energy* 323 (2022), p. 119603. ISSN: 0306-2619. DOI: https://doi.org/10.1016/j.apenergy.2022.119603. URL: https://www.sciencedirect.com/science/article/pii/S0306261922009102.

[29] J. Wettengel. "Volatile but predictable: Forecasting renewable power generation". In: *Clean Energy Wire* (Aug. 2016). URL: https://www.cleanenergywire.org/factsheets/volatile-predictable-forecasting-renewable-power-generation.

[30] R. H. Inman, H. T. C. Pedro, and C. F. M. Coimbra. "Solar forecasting methods for renewable energy integration". In: *Progress in Energy and Combustion Science* 39.6 (2013), pp. 535–576.

[31] Marco Peixeiro. *The Complete Guide to Time Series Analysis and Forecasting*. https://towardsdatascience.com/the-complete-guide-to-time-series-analysis-and-forecasting-70d476bfe775. Accessed May 26, 2023.

[32] Wikipedia. *Moving average*. https://en.wikipedia.org/wiki/Moving_average. Accessed: May 26, 2023.

[33] Jacopo De Stefani. *Towards multivariate multi-step-ahead time series forecasting: A machine learning perspective*. Ph.D. thesis, Advisor: Prof. Gianluca Bontempi. Feb. 2022. URL: https://www.researchgate.net/publication/359031751_Towards_multivariate_multi-step-ahead_time_series_forecasting_A_machine_learning_perspective.

[34] Steven L. Brunton and J. Nathan Kutz. *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge University Press, 2019, p. 119. ISBN: 978-1-108-42209-3.

[35] Jacopo De Stefani. *Towards multivariate multi-step-ahead time series forecasting: A machine learning perspective*. Thesis. Ph.D. thesis, Advisor: Prof. Gianluca Bontempi. Feb. 2022. URL: https://www.researchgate.net/publication/359031751_Towards_multivariate_multi-step-ahead_time_series_forecasting_A_machine_learning_perspective.

[36] Wikipedia. *Decision tree learning*. Accessed: May. 5, 2023. May 2023. URL: https://en.wikipedia.org/w/index.php?title=Decision_tree_learning&oldid=1157731697.

[37] Wikipedia. *Random forest*. Accessed: May. 5, 2023. June 2023. URL: https://en.wikipedia.org/w/index.php?title=Random_forest&oldid=1160091469.

[38] Jason Brownlee. *How to Use XGBoost for Time Series Forecasting*. Accessed: May 26, 2023. Aug. 2020. URL: https://machinelearningmastery.com/xgboost-for-time-series-forecasting/.

[39] Yuanchao Wang, Z. Pan, J. Zheng, L. Qian and Li Mingtao. "A hybrid ensemble method for pulsar candidate classification". In: *Astrophysics and Space Science* 364 (Aug. 2019). DOI: 10.1007/s10509-019-3602-4.

[40] Jason Brownlee. *CNN Long Short-Term Memory Networks*. Accessed: May 28, 2023. Aug. 2017. URL: https://machinelearningmastery.com/cnn-long-short-term-memory-networks/.

[41] Jason Brownlee. *How to Develop a Bidirectional LSTM For Sequence Classification in Python with Keras*. Accessed: May 28, 2023. June 2017. URL: https://machinelearningmastery.com/develop-bidirectional-lstm-sequence-classification-python-keras/.

[42] Yunghui Li, Latifa Nabila Harfiya, Kartika Purwandari and Yue-Der Lin. "Real-Time Cuffless Continuous Blood Pressure Estimation Using Deep Learning Model". In: *Sensors* 20 (Sept. 2020), p. 5606. DOI: 10.3390/s20195606.

[43] F. Wyffels and B. Schrauwen. "A comparative study of Reservoir Computing strategies for monthly time series prediction". In: *Neurocomputing* 73.10 (2010). Subspace Learning / Selected papers from the European Symposium on Time Series Prediction, pp. 1958–1964. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2010.01.016. URL: https://www.sciencedirect.com/science/article/pii/S0925231210000962.

[44] Eric Antonelo and Benjamin Schrauwen. "Supervised Learning of Internal Models for Autonomous Goal-Oriented Robot Navigation using Reservoir Computing". In: June 2010, pp. 2959–2964. DOI: 10.1109/ROBOT.2010.5509212.

[45] Wikipedia. "Hyperparameter Optimization". In: (2023). Accessed: May 28, 2023. URL: https://en.wikipedia.org/wiki/Hyperparameter_optimization.

[46] Wikipedia. *Metaheuristic*. https://en.wikipedia.org/wiki/Metaheuristic. Accessed: May 28, 2023.

[47] Oussama Merabet et al. "Optimal coordination of directional overcurrent relays in complex networks using the Elite marine predators algorithm". In: *Electric Power Systems Research* 221 (2023), p. 109446. ISSN: 0378-7796. DOI: https://doi.org/10.1016/j.epsr.2023.109446. URL: https://www.sciencedirect.com/science/article/pii/S0378779623003358.

[48] Hamza Belmadani et al. “A twofold hunting trip African vultures algorithm for the optimal extraction of photovoltaic generator model parameters”. In: *Energy Sources, Part A: Recovery, Utilization, and Environmental Effects* 44.3 (2022), pp. 7001–7030. DOI: 10.1080/15567036.2022.2025265.

[49] Dyhia Bouhadjra, Aissa Kheldoun, and Ali Zemouche. “Performance analysis of stand-alone six-phase induction generator using heuristic algorithms”. In: *Mathematics and Computers in Simulation* 167 (2020). INTERNATIONAL CONFERENCE on Emerging and Renewable Energy: Generation and Automation, held in Belfort, France on 4-6 July, 2017, pp. 231–249. ISSN: 0378-4754. DOI: https://doi.org/10.1016/j.matcom.2019.06.011. URL: https://www.sciencedirect.com/science/article/pii/S0378475419302095.

[50] Wikipedia. *Differential Evolution*. https://en.wikipedia.org/wiki/Differential_evolution. Accessed: May 30, 2023.

[51] G. Jeyakumar and C. Shanmugavelayutham. “Convergence Analysis of Differential Evolution Variants on Unconstrained Global Optimization Functions”. In: *Internation Journal of Artifical Intelligence and Applications* (2011). DOI: 10.5121/ijaia.2011.2209. URL: https://arxiv.org/pdf/1105.1901.

[52] Adrian Tam. *A Gentle Introduction to Particle Swarm Optimization*. Machine Learning Mastery. Sept. 2021. URL: https://machinelearningmastery.com/a-gentle-introduction-to-particle-swarm-optimization/.

[53] Wikipedia. *Particle Swarm Optimization*. accessed: May 30, 2023. URL: https://en.wikipedia.org/wiki/Particle_swarm_optimization.

[54] Vitor Cerqueira. *4 Things to Do When Applying Cross-Validation with Time Series*. Accessed: May 30, 2023. URL: https://towardsdatascience.com/4-things-to-do-when-applying-cross-validation-with-time-series-c6a5674ebf3a.

[55] Venishetty Vineeth, Huseyin Kusetogullari, and Alain Boone. “Forecasting Sales of Truck Components: A Machine Learning Approach”. In: May 2020, pp. 510–516. DOI: 10.1109/IS48319.2020.9200128.

[56] *DKA Solar Centre - Alice Springs*. Accessed: June 5, 2023. URL: https://dkasolarcentre.com.au/locations/alice-springs.

[57] *DKA Solar Centre - Alice Springs, Sanyo, 6.3kW, HIT Hybrid Silicon, Fixed, 2010*. Accessed on June 5, 2023. URL: https://dkasolarcentre.com.au/source/alice-springs/dka-m4-b-phase.

[58] Desert Knowledge Australia Centre. *Download Data: 17 Sanyo, 6.3kW, HIT Hybrid Silicon, Fixed, 2010*. Online. Accessed: April 25, 2023. URL: http://dkasolarcentre.com.au/download.

[59] A. A. A. AL-qamadi. "Etude de la centrale photovoltaïque de Dhaya 'Sidi Bel Abbés'". MA thesis. Université Mohamed Boudiaf - M'Sila, 2020.

[60] Wikipedia. *Box plot*. Accessed: June 6, 2023. June 2023. URL: https://en.wikipedia.org/wiki/Box_plot.

[61] Jihoon Moon, Jinwoong Park and Eenjun Hwang. "Forecasting power consumption for higher educational institutions based on machine learning". In: *J Supercomput* 74.8 (Aug. 2018), pp. 3778–3800. DOI: 10.1007/s11227-017-2022-x.

[62] PL Bescond. *Cyclical features encoding, it's about time!* Accessed on 03 June 2023. June 2020. URL: https://towardsdatascience.com/cyclical-features-encoding-its-about-time-ce23581845ca.

[63] Tirthajyoti Acharya. *Understanding Feature extraction using Correlation Matrix and Scatter Plots*. Accessed: June 9, 2023. Towards Data Science. Mar. 2020. URL: https://towardsdatascience.com/understanding-feature-extraction-using-correlation-matrix-and-scatter-plots-6c19e968a60c.