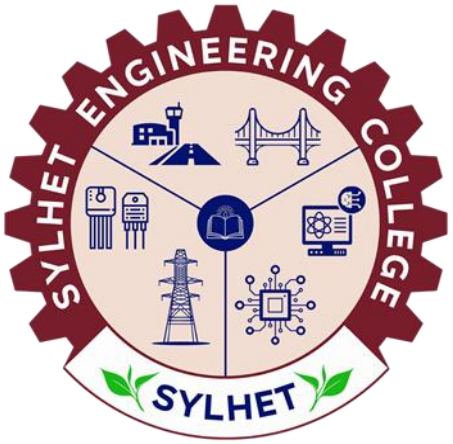
A Thesis Submitted to the Sylhet Engineering College for the Degree of
Bachelor of Science in Electrical and Electronic Engineering

Utilizing Sky Image Dataset for Short-Term Solar forecasting by using Hybrid CNN-LSTM and ConvLSTM-LSTM Models

By
Ahmed Thawhid Sabit
&
Rayhan Ahmed Opi

Supervised by,
Md. Ashraful Alam
Lecturer
Department of Electrical and Electronic Engineering
Sylhet Engineering College



March, 2024
Sylhet Engineering College, Sylhet
Affiliated with
Shahjalal University of Science & Technology (SUST)

The thesis titled **"Utilizing Sky Image Dataset for Short-Term Solar forecasting by using Hybrid CNN-LSTM and ConvLSTM-LSTM Models"** submitted by **Ahmed Thawhid Sabit**, **Rayhan Ahmed Opi**; Student ID: **2019338544**, and **2019338515**; Session **2019-2020**, to the Department of Electrical and Electronic Engineering, Sylhet Engineering College, has been accepted as satisfactory in partial fulfillment of the requirement for the Degree of Bachelor of Science in Electrical and Electronic Engineering and approved as to its style and contents.

BOARD OF EXAMINERS

_____
Md. Shahid Iqbal
Assistant Professor and Head
Department of Electrical and Electronic Engineering
Sylhet Engineering College, Sylhet.

Chairman

_____
Salman Fazle Rabby
Assistant Professor
Department of Electrical and Electronic Engineering
Sylhet Engineering College, Sylhet.

Member

_____
Apurba Biswas
Assistant Professor
Department of Electrical and Electronic Engineering
Sylhet Engineering College, Sylhet.

Member

_____
Md. Ashraful Alam
Lecturer
Department of Electrical and Electronic Engineering
Sylhet Engineering College, Sylhet.

Member & Supervisor

_____
Mahedi Kamal Ahmed
Lecturer
Department of Electrical and Electronic Engineering
Sylhet Engineering College, Sylhet.

Member

_____
Arif Ahammad
Assistant Professor
Department of Electrical and Electronic Engineering
Shahjalal University of Science & Technology, Sylhet

Member (External)

Acknowledgements

We start by expressing our appreciation to the Almighty, whose guidance and support have allowed us to pursue this journey and fulfill the requirements for the Bachelor of Science in Electrical and Electronics Engineering (EEE), including our thesis work. It is by His grace that we have gained the strength and resolve to reach this significant milestone.

We would like to take this opportunity to convey our deep gratitude to our honorable supervisor, **Md. Ashraful Alam**, Lecturer at the Department of EEE at Sylhet Engineering College, for his insightful academic counsel, innovative guidance, ongoing encouragement, valuable recommendations, and all the other support he has provided throughout our studies. His exceptional supervision has greatly benefited us. Additionally, it is an honor to express our gratitude to our head, **Md. Shahid Iqbal**, and faculty member **Salman Fazle Rabby** for their generous assistance and collaboration throughout this journey. We would also like to extend our thanks to Md. Janibul Alam Soeb, Assistant Professor, Department of FPM (Farm Power and Machinery), Faculty of Agricultural and Engineering Technology, Sylhet Agricultural University. Their insightful discussions and support during this thesis have left us deeply appreciative of his contributions.

We would like to express our sincere appreciation to our parents, siblings, classmates, and friends in the EEE department at SEC. We recognize the sacrifices, prayers, motivation, and steadfast support they have given us, which have played a crucial role in our achievements. We are grateful to them for influencing our academic and personal growth.

Abstract

This thesis presents a comprehensive study on present time nowcasting and short-term solar radiation forecasting using hybrid deep learning architectures that integrate Convolutional Neural Networks (CNN), Convolutional Long Short-Term Memory (ConvLSTM), and LSTM models. The purpose of this study is to effectively forecast the PV value using ground-based sky image dataset sourced from Stanford University. This research employs various phases, including historical data analysis, data processing, spatial and temporal feature extraction, CNN-LSTM and ConvLSTM-LSTM model development and hardware implementation. The study develops and evaluates nowcasting and forecasting models, where CNN and ConvLSTM layers extract spatiotemporal features (e.g. cloud cover analysis) from sky images, while LSTM layers capture temporal dependencies for accurate solar PV generation prediction. In addition to the models, real-time meteorological parameters were collected via a custom-built Internet of Things (IoT) system, and sky image data were captured using a ground-based camera system.Experimental results demonstrate that the hybrid CNN-LSTM nowcasting model achieved a Mean Squared Error (MSE) of 6.08 W/m², Root Mean Squared Error (RMSE) of 2.47 W/m², Mean Absolute Error (MAE) of 1.34 W/m², and an R-squared value of 0.896, indicating strong predictive performance. The ConvLSTM-LSTM forecasting model attained an MSE of 7.40 W/m², RMSE of 7.40 W/m², MAE of 1.45 W/m², and an R-squared value of 0.873, demonstrating robust forecasting capability under dynamic weather conditions. These findings highlight the potential of sky image analysis for enhancing short-term solar power forecasting, thereby contributing to improved grid stability, reduced reliance on fossil fuel backups, and more efficient renewable energy integration.

Keywords: *Sky image, Spatiotemporal , spatial, temporal, nowcast, forecast, short-term prediction, hybrid deep learning.*

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction

1.1 Overview

Solar energy is derived from the sun's radiation, created by nuclear fusion occurring in its core. This renewable energy source is plentiful and can be used in multiple ways, such as generating electricity and providing heating. As the world grapples with the challenges of fossil fuel depletion and climate change, solar energy emerges as a clean, renewable alternative that can significantly contribute to energy security and environmental sustainability.

Predicting solar energy production is essential for maximizing the efficiency of solar power, especially with the rising global adoption of photovoltaic (PV) systems. It encompasses forecasting solar power output using a variety of data sources and techniques, which can greatly improve the effectiveness and dependability of solar energy systems. Recent advancements have seen the integration of AI and machine learning, which improve forecast accuracy by analyzing large datasets and identifying patterns [1][2]. There are several methods to anticipate solar power such as the nowcasting and short-term method, and the long-term method. The nowcasting methods utilize real-time data from ground sensors and satellite imagery to predict solar power generation in the immediate future [3]. And the long-term approach relies on historical weather patterns and climate models to estimate solar energy production over extended periods [3].

Solar forecasting using image processing is an innovative approach that leverages visual data to predict solar irradiance and optimize energy management in solar power systems. By analyzing images captured from various sensors, researchers can extract critical information about cloud cover and atmospheric conditions, which directly influence solar energy generation. Deep learning approaches such as Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU) have shown exceptional accuracy in forecasting solar energy production, with LSTM achieving the lowest RMSE in comparative studies [4]. This method enhances the accuracy of short-term forecasts, enabling better energy dispatch and grid stability.

Solar forecasting using sky images is an innovative approach that leverages ground-based camera systems to capture real-time cloud patterns and atmospheric conditions. By analyzing these images with advanced machine learning models, such as CNNs and LSTMs, it becomes possible to accurately predict short-term solar irradiance and power generation. This method offers significant advantages over traditional numerical weather models by providing high-resolution, localized forecasts. Therefore, the key contribution of this study focuses on.

- Training Deep learning models by extracting spatial and temporal features from open-source SKIPP'D dataset.
- Implementing CNN model for extracting spatial for nowcast P power generation for present time.
- Using ConvLSTM with LSTM layers for robust short-term forecast model.

1.2 What is Solar Forecasting?

Solar forecasting plays a crucial role in planning and optimizing solar energy generation and energy trading. Solar forecasting is a procedure used to predict the amount of sunlight or solar radiation reaching Earth's surface. It is also used to predict photovoltaic (PV) systems' electricity generation. Solar forecasting enables the effective incorporation of solar energy into smart grids, which enhances power distribution and reliability [5]. Forecasting of solar power is also necessary for energy grid operators, utility companies, and also for making renewable energy. Solar forecasting depends on some major terms, such as solar irradiance, temperature, humidity, wind, latitude, time of the year, and sun angle etc.

Recently solar forecasting delivered promising output through sky image data compared to numerical data for short term solar forecasting. Image-based data approaches, especially those employing deep learning techniques, have demonstrated encouraging outcomes in short-term solar forecasting by monitoring real-time cloud movements. On the other hand, numerical data, typically sourced from numerical weather predictions (NWPs), offers a wider atmospheric context but may fall short in providing the spatial resolution necessary for precise localized solar energy forecasts.

1.3 Objectives of Solar Forecasting

Solar forecasting is significant for power generation and increasing the reliability of the power grid. The objectives are given below:

- Generally, the solar radiation differs due to the weather and other parameters. Solar forecasting helps the grid operators to keep the power supply balanced and stable.
- Solar forecasting increases the performance and maintenance quality of the grid by predicting solar power.
- It can reduce the fossil fuel-based backup power sources by forecasting solar energy.
- It helps to determine the size of the grid, gives an analytical idea about the expansion of the grid by providing long-term data of solar power. It also assists in making decisions on investment in the power grid.
- Reduce the operational and maintenance costs by providing the prediction of output data.
- Solar forecasting increases the solar plant's production accuracy, which makes the power plants more reliable.
- Precise solar forecasting provides accurate load forecasting by calculating solar power generation.

1.4 Types of Solar Forecasting

Solar forecasting is not only necessary for the next 1/2 hour but also essential for a week, a month, or some years. To fulfill this necessity, solar forecasting is divided into four types depending on time zone:

- **Nowcasting**

Nowcast forecasting is a kind of forecasting that refers to real-time or nearly real-time anticipation of solar power radiation. It may fluctuate from minutes to a few hours or sometimes up to a day. In general, nowcasting predicts only succeeding 0-6 hours of solar Radiation.

Nowcast forecasting is very essential for fog predictions at airports, rainfall density checking, and thunderstorm warnings. This process also helps to estimate the GDP. Public health and the financial market by using real-time data.

- **Short-Term Forecasting**

Short-term forecasting refers to predicting solar power radiation for a few hours to a few weeks. It is broadly used in industries for planning and operational decisions. Short-term forecasting

helps to predict the temperature, snow, and rain for 1-7 days. Solar and Wind energy are also predicted by this method to adjust the grid maintenance. This method is used to predict for a short period, which may help agriculture, health, airlines, and so on. Decision Tree, Random Forest, Gradient Boosting, Neural Networks, especially LSTM, etc, are used as machine learning models for predicting short-term forecasting.

- **Mid-Term Forecasting**

Mid-term forecasting refers to predicting the amount of solar energy for the next 2-15 days or sometimes up to months. Mid-term forecasting is usually applied in the power grid planning and maintenance. It also helps to improve planning on solar power plants and reduces the dependency on fossil fuels. Time series models, LSTM, Random forest, etc, are utilized as machine learning models for forecasting. Cloud cover data is difficult to predict in a mid-term forecasting method.

- **Long Term Forecasting**

Long-term forecasting means predicting the solar power for a period ranging from some months to several years. In long-term forecasting, investment decisions and policy-making can be utilized. It helps to inform how much solar energy is produced in a year in a particular area, which may help to make decisions about whether building a solar plant is effective or worthless. It may face difficulties due to seasonal variations and needs an enormous amount of historical data to train a model. ANN (Artificial Neural Networks), RNN (Recurrent Neural Networks), SVR (Support Vector Regression), LSTM, etc, are used to predict solar energy through long-term forecasting.

1.5 Importance of Solar Forecasting

Solar forecasting is effectively implemented in the power system by predicting solar energy. It helps to utilize solar energy efficiently in the power system for maintaining the grid. Solar forecasting is dependent on weather, but accurately forecasting increases better planning and cost management.

Basically, solar power fluctuates with changes of season and time, which may cause less accuracy in the prediction. But if it is enabled to maintain the accuracy, then it can provide grid

operators with a balanced supply and reduce the risk of blackouts. Solar forecasting plays an effective role in grid operators. It saves power from overgeneration and under generation by predicting solar power accurately. Proper solar forecasting is penetrating the solar power into the power system, which makes a balanced and beneficial energy mix system. Since solar is a renewable source, it will provide pollution-free fuel to the grid and save the cost of fossil fuel. On the other hand, solar forecasting helps in multiple functions such as return on investment calculation, financial planning on the grid, planning renewable targets, load prediction, energy budgeting, and so on.

1.6 Objectives of This Study

- To develop robust hybrid deep learning models (CNN-LSTM and ConvLSTM-LSTM) that can effectively extract both spatial and temporal features from sky image datasets for short-term solar forecasting.
- To investigate the capability of sky images as predictive features for solar irradiance forecasting, in comparison to conventional numerical or meteorological datasets.
- To design and evaluate a CNN-LSTM framework that captures spatial cloud patterns through convolutional layers and learns sequential dependencies through recurrent layers.
- To design and evaluate a ConvLSTM-LSTM framework that directly integrates spatiotemporal dependencies from sky images, aiming to improve forecasting accuracy.
- To compare the performance of the proposed hybrid models in terms of forecasting accuracy, error metrics (MAE, RMSE, MAPE, R^2), and computational efficiency.

1.8 Thesis Structure

This study is organized into seven chapters, which represent the research on solar forecasting by using a Hybrid CNN-LSTM and ConvLSTM-LSTM models.

Chapter 1: Introduction

This chapter presents the background, motivation, problem statement, and objectives of the research, highlighting the importance of short-term solar forecasting using sky images.

Chapter 2: Literature Review

A critical review of existing solar forecasting methods, including statistical, machine learning, and deep learning approaches, with special focus on hybrid CNN-LSTM and ConvLSTM-based models.

Chapter 3: Theoretical Analysis

This chapter explains the theoretical foundations of convolutional neural networks (CNNs), long short-term memory networks (LSTMs), and ConvLSTMs, outlining their suitability for spatiotemporal forecasting tasks.

Chapter 4: Dataset

Details about the sky image dataset, preprocessing steps, feature extraction, and data preparation techniques required for training and evaluating the proposed models.

Chapter 5: Methodology

This chapter describes the proposed hybrid CNN-LSTM and ConvLSTM-LSTM architectures, model design, implementation strategies, and training procedures.

Chapter 6: Evaluation Metrics and Results Analysis

Presentation of performance evaluation criteria (MAE, RMSE, MAPE, R^2) followed by a comparative analysis of experimental results to assess the effectiveness of the proposed models.

Chapter 7: Conclusion and Future Work

A summary of key findings, contributions, and limitations of the research, along with potential directions for future advancements in solar forecasting using deep learning and sky images.

In summery, The introduction of this thesis has outlined the fundamental aspects of solar forecasting and its significance in the modern energy landscape. It began with an overview of solar forecasting, defining its role in predicting solar irradiance to ensure efficient integration of renewable energy into power systems. The objectives of solar forecasting were discussed, emphasizing its importance in enhancing grid reliability, reducing energy imbalance, and optimizing solar energy utilization. Various types of solar forecasting were introduced, with a particular focus on short-term forecasting due to its critical role in handling rapid cloud variations and maintaining real-time grid stability. Finally, the importance of solar forecasting was highlighted in terms of economic, technical, and environmental benefits, which collectively establish the foundation for this research.

Chapter 2: Literature Review

Accurate solar forecasting provides more reliability and efficiency of the renewable energy system. Many investigations reveal multiple ways to solar forecasting by using cloud coverage data or numerical data. But comparing both datasets, cloud cover data supply improved forecasting by analyzing weather conditions and delivering reliable output. Different thesis papers utilized different kinds of Machine Learning and Deep Learning models like ANN, CNN, ARIMA, LSTM, SVR, Regression type Model, etc, for both numerical and cloud cover data. Most of the paper uses numerical data from many open sources. But cloud coverage data shows better output and reliable results than numerical datasets.

The literature review represents a comparative overview of the latest relevant papers that can help a researcher achieve their goal. Here, this studies explore some research gaps and provides an overall comprehensive overview of recent papers:

Yuhau Nie et al.[6] introduces SKIPP'D, a specialized dataset comprising sky images and PV power generation data aimed at advancing short-term solar forecasting through deep learning approaches i.e. CNN. The paper aims to facilitate reproducible research, model comparison, and development of new forecasting methods. The researchers compiled and released a rich dataset spanning three years (2017–2019), including both benchmark (64×64 resolution) and raw (2048×2048 resolution) sky images paired with 1-minute resolution PV power generation data. The images were captured using a 360-degree fisheye camera installed at Stanford, and PV data were logged from nearby rooftop solar panels. The model CNN named it SUNSET, shows the RMSE and MAE are low (0.61 kW and 0.50 kW, respectively) for sunny days, but performance degrades under cloudy conditions (RMSE 4.27 kW, MAE 2.95 kW). The model tends to predict conservatively in volatile weather, failing to capture sharp fluctuations. The authors recommend further research on probabilistic models and improved forecasting of cloud movement to address these issues.

Suraj Chand et al. [7] address the issue of accurately forecasting solar radiation, which is essential for enhancing grid stability and efficient solar energy utilization in New Delhi, India. The study proposes a hybrid model combined with a Random Forest Regression model and Long Short-Term Memory (LSTM) networks to anticipate the Global Horizontal Irradiance (GHI) of the next 24 hours. They collect meteorological data from NREL for 6 years

(2015-2020), consisting the weather parameters such as temperature, humidity, and sky clearness index. This study was limited to a single location, i.e, New Delhi, and used only 6 years of data, which is an insufficient dataset for training a model. And also, they are not enabled to use real-time data. The hybrid model achieved 98% accuracy with lower error metrics such as MAE (26.22 W/m²), MSE (1304.45 W/m²), and RMSE (36.11 W/m²).

The study by Rial A. Rajagukguk et al. [8] investigates a novel approach to short-term solar irradiance forecasting by capturing sky image data through a hemispherical sky camera and deep learning techniques at a fixed location, i.e, Kookmin University in Seoul, South Korea. The paper captured one image per minute and paired it with Global Horizontal Irradiance (GHI), which is measured by a pyranometer. Approximately 1200 sky images were used to estimate the cloud cover using the Red-Blue Ratio(RBR) method, which was then utilized in Long Short Term Memory(LSTM) to forecast cloud cover for the next 10 minutes. The forecasted data was applied to a modified Kasten physical solar radiation model to predict GHI. The proposed hybrid method performed better than the Yoo and Persistence model under cloudy conditions, achieving a root mean square difference (RMSD) of 199.75 W/m², significantly lower than the persistence model's 317.94 W/m². This study focused on a single location, and the applied model lacks real-time adaptability.

Seongha Park et al. [9] propose a machine learning-based framework for predicting solar irradiance and photovoltaic energy output by estimating cloud coverage from a sky-facing camera. The authors utilize a combination of datasets- SWIMSEG, HYTA, and Waggle cloud dataset, containing diverse sky conditions, to train and validate their models. The authors employed three deep learning models, such as U-Net, FCN, and DeepLab v3, and a color-based Partial Least Squares (PLS) regression model, and two ensemble AdaBoost models. They found that U-Net exhibited the highest cloud segmentation accuracy among them, and it was mIoU = 0.7626, mAP = 0.9869, mAR = 0.7703, but DeepLab v3 shows the lowest ME, i.e, 0.22 during clear sky. The study was not able to differentiate the thickness of the cloud, so the output varies while the cloud condition changes.

Yuhao Nie et al. [10] introduce SkyGPT, a novel two-stage deep learning model for short-term solar forecasting by using sky videos. The system first generates multiple future sky images

via a two-video prediction model combining VideoGPT and PhyDNet, then maps those images to PV power output using a modified U-Net model. The study collects data from Stanford University, consisting of over 334,000 paired samples of sky images and PV power data from a

rooftop 30 kW solar system. The model achieved superior performance compared to benchmarks with a continuous ranked probability score of 2.81 by coupling SkyGPT and U-Net and a Winkler score of 26.70, outperforming SUNSET and smart persistence models by 13–23% in forecast skill. The performance of the system depends on the accuracy of video prediction; lower precision is required under extreme cloud dynamics. The study suggests future work could focus on enhancing video prediction accuracy.

Yuhao Nie, Xiatong Li, and others [11] conduct a comprehensive survey focusing on open-source ground-based sky image datasets for very short-term solar forecasting, cloud analysis, and modeling. Their main goal was to tackle the shortage of data, which currently hinders how well deep learning algorithms can predict solar energy, categorize cloud types, separate clouds from images, and follow cloud movement. The study identifies and evaluates 72 publicly available datasets using a structured multi-criteria ranking system encompassing dimensions such as data resolution, spatial and temporal coverage, accessibility, and quality control. These collections of data come from various sources, including government science bodies and university research teams, spanning different climate regions and containing diverse information like actual sky photos, measurements of solar radiation, visual maps showing segmented clouds, and weather data. While the paper highlights the growing availability of such datasets as a positive trend, it also notes significant limitations. These include inconsistent data quality across sources, limited annotations for supervised learning, and variability in temporal resolution that affects suitability for time-sensitive forecasting tasks. The authors emphasize that the lack of standardized labeling and varying degrees of documentation hinder the broad applicability of some datasets.

Yuhao Nie, Quentin Paletta, and others [12] investigate the development of deep learning-based solar forecasting models utilizing sky images collected from three different geographic locations: Stanford, SIRTA (Palaiseau, France), and DEWA (Dubai, UAE). This study

compares three training strategies: training machine learning models locally on individual datasets, training a combined global model on fused datasets, and transfer learning, where data is transferred from pre-trained models to new datasets. This paper aims to evaluate the effectiveness of prediction accuracy and training efficiency, addressing challenges caused by data variety, including differing scales, distributions, and camera setups. The models employed include the SUNSET CNN architecture and ConvLSTM, with performance measured via RMSE and computational effort. The results reveal that local models excel when applied within their origin sites, but their transferability across locations suffers in terms of accuracy. On the other

hand, global models trained on fused datasets demonstrate robust generalization and significantly reduce training cost while maintaining high accuracy. This paper shows dataset heterogeneity and computational resource demands for training global models.

Q. Paletta et al. [13] researched the use of deep learning techniques, specifically CNNs, to improve short-term solar irradiance predictions, utilizing data from ground-based sky cameras. The researchers constructed a hybrid architecture that integrated Convolutional Neural Networks (CNNs) for extracting visual features from sky images with Artificial Neural Networks (ANNs) for handling supplementary data. This development relied on over 64,000 sky images and Global Horizontal Irradiance (GHI) data collected at the SIRTA atmospheric research site in France from February to September 2018. The model achieved a forecast skill score of up to 44% for 10-minute-ahead predictions. Its performance improved notably when trained on the same-day data, reflecting a realistic operational setting. The model is not able to anticipate sudden irradiance changes, and the architecture may not leverage past irradiance measurements effectively, which could enhance the model accuracy.

Anas Al Lahham et al. [14] focus on the advancements of solar forecasting using machine learning, like KNN and Random Forest(RF), which enhances the accuracy of PV systems. The major aim of the authors is to improve the efficiency of forecasting by implementing a short-term method. The authors used two publicly available datasets of sky images, such as Total Sky Imager (TSI) and All Sky Imager (ASI). The study utilizes the datasets containing over 350,000 sky images collected over 16 years (2004-2020), with corresponding global horizontal irradiance (GHI) values serving as ground truth for model evaluation. The paper proposed that a KNN-based model was achieved for TSI and ASI- 16 hours forecast, the

nMAPE is 14.9% and 14.7%, the RMSE is 122.2 W/m2 and 116.7 W/m2, and the nRMSE is 7.7% and 8.9%, respectively. This model increases complexity and computational time when multiple inputs are used. The method is only applicable for short-term forecasting time ranging from 1-4 hours, which may not be able to anticipate long-term forecasting, and the model performance varies significantly under different weather conditions.

Elissaios Alexios Papatheofanous et al. [15] address the critical challenge of solar irradiance forecasting, which is significant for optimizing photovoltaic(PV) power production. The researchers reveal the lack of a typical solar forecasting method, which often struggles with the changes in cloud conditions. To overcome this problem, the study applied Convolutional Neural Networks (CNN) and image processing techniques, especially focusing on sun localization to

increase the forecasting accuracy. For the development of the CNN model, the researchers utilized the Folsom, CA dataset available publicly. The dataset contains the high-resolution 1536 X 1536 sky images taken at Folsom, CA, USA, collected over three years from 2014-2016. The major objectives of this study are to improve the accuracy of solar forecasting using a CNN model, thereby enabling real-time control of the PV system. The authors proposed to improve CNN's result and enhance accuracy for solar irradiance estimation by applying the SunMask generation image processing method. They evaluate the various CNN models like VGG11, ResNet-50, MobileNetV2, and SqueezeNet, which reveals that the SunMask method.The ResNet-50 model appears to result in the lowest error metrics, with an RMSE of 64.83 W/m2 . The MobileNetV2 model is favoured the most by our SunMask generation method, with its RMSE decreasing from 75.95 W/m2 to 65.51 W/m2 , a 13.75% improvement. This papers only utilize the data of Folson and CA, which may not represent other region and their different climate condition. The authors applied CNN, which may lead to complexity of the irradiance forecasting, and they acknowledge that future work can be more beneficial by exploring advanced deep learning architecture, i.e, LSTM.

Wen-Chi Kuo et al. [16] present a novel approach to boost the accuracy of photovoltaic(PV) power forecasting by employed cloud data taken from sky images using various machine learning models, including Artificial Neural Networks (ANN), Long Short-Term Memory (LSTM), and Gated Recurrent Units (GRU), to compare their forecasting capabilities under different weather conditions. This study aims to evaluate the effectiveness of cloud coverage

as a critical weather factor in improving short-term PV power estimation. The authors assemble the data from February 21, 2021, to June 13, 2021, for analysis and forecasting purposes. The researchers focus on processing sky images using RGB pixels and utilize the OpenCV library in Python for analyzing the RGB distribution. The results show that the application of the deep learning models LSTM and GRU is more effective than ANN for short-term PV power estimation. However, the anticipation result illustrates that the system is invalid for rainy weather conditions and the model depends on specific weather conditions. Also, the result shows that the ANN model performs well in sunny weather, while its output is poor in cloudy or rainy weather.

N.Y Hendrikx et al. [17] develops a 20-minute-ahead forecasting model using LSTM networks and shows the comparison of the LSTM model against Random Forest and ANN. The research involves extracting features from ASI(All Sky Images)—such as cloud pixel counts, brightness, and edges. These features are used to train and evaluate multi-output LSTM models, which are benchmarked against Random Forest (RF), Artificial Neural Networks (ANN). Data were collected from two ASI cameras in southern Spain over a five-month period. LSTM achieves notable Forecast Skill Scores of 39% in sunny and 25% in partially cloudy conditions. It performs in partially cloudy conditions, with an RMSE of 76.83, MAE of 43.12, and a Ramp-score of 29.04. The future work includes incorporating more cloudy-day data, testing advanced cloud segmentation methods.

Sujan Ghimire et al. [18] presents a novel hybrid deep learning model (CLSTM) for short-term forecasting. The primary objective of the study is to accurately predict half-hourly GSR by integrating the feature extraction capability of CNN with the temporal sequence learning ability of LSTM networks. The study highlights the robustness and accuracy of the hybrid model (CNN-LSTM) in both short- and mid-term forecasting. Using data from Alice Springs, Australia (2006–2018), the authors constructed and validated the CLSTM model against some models including CNN, LSTM, RNN, GRU, DNN, MLP, and Decision Trees, over forecast horizons ranging from 1 day to 8 months. The paper showed that the CLSTM model significantly outperformed all comparison models across all metrics, with over 70% of prediction errors within ±10 W/m² and a 1-day prediction RRMSE of approximately 1.52%, MAPE of 4.67% and RMAE 6.305%. However, it is limited to a single location and uses only solar radiation data, which restricts generalizability.

2.1 Research Gap

Recent studies have increasingly emphasized the usefulness of whole sky images as a powerful tool for capturing real-time cloud dynamics, since they provide continuous visual information about cloud formation, movement, and dissipation. These dynamics directly influence solar irradiance, making sky images an invaluable input for forecasting solar power generation. Despite this, the majority of existing research has focused on using standalone architectures—such as Convolutional Neural Networks (CNNs) for spatial feature extraction, Random Forests (RF) for decision-based predictions, or ConvLSTMs for handling spatio-temporal sequences. While each of these models contributes valuable insights, relying on a single architecture limits their ability to fully capture the complex interplay between spatial patterns, temporal evolution, and atmospheric variability. This highlights a significant gap: the underutilization of hybrid approaches that could leverage the complementary strengths of different models. By combining spatial, temporal, and contextual learning mechanisms within a unified framework, hybrid architectures hold the potential to achieve more accurate and robust solar forecasting compared to traditional standalone methods.

In summery, The literature review highlights the importance of accurate solar forecasting in improving the efficiency and reliability of renewable energy systems. The review shows that while many studies have used numerical weather data, cloud coverage data from sky images provides more reliable and accurate forecasts due to its direct link with solar irradiance variations. A wide range of machine learning and deep learning models, such as CNN, LSTM, ConvLSTM, ANN, RF, ARIMA, and hybrid models, have been applied across different studies, each with strengths and limitations. Several researchers introduced specialized datasets, such as SKIPP'D and other ground-based sky image collections, to support reproducible research and model comparison. Key findings from the reviewed works indicate that hybrid models combining spatial and temporal learning outperform standalone approaches, and models trained on global datasets tend to generalize better than those limited to local datasets. However, challenges remain, including performance degradation under cloudy or rapidly changing weather, lack of standardized datasets, limited adaptability across regions, and insufficient real-time capability. Overall, the literature review identifies the growing potential of image-based and hybrid deep learning methods for short-term solar forecasting, while also pointing out existing research gaps that motivate further exploration.

Chapter 3: Theoretical Analysis

3.1 Hardware Components

A custom-built IoT-based system was developed for real-time weather data acquisition. The system integrated multiple sensors and modules, including Arduino Uno, ESP32, Anemometer, BMP280 (barometric pressure), DHT22 (temperature and humidity), Rain Gauge, Wind Vane, RTC module (for time and date stamping), and a fishing camera for sky image capture. Each component played a specific role in measuring key environmental parameters such as wind speed, wind direction, atmospheric pressure, humidity, temperature, and rainfall.

3.1.1 Arduino Uno

The Arduino Uno is a versatile microcontroller widely utilized in various applications due to its user-friendly programming environment and robust capabilities. It is powered by the ATMega328P Integrated Circuit and the operating voltage is 5V that offers an accessible programming environment, allowing users to easily write and upload code. It can manage multiple input and output devices, making it ideal for controlling motors, sensors, and other components in various projects. It is employed in smart parking solutions, utilizing ultrasonic sensors to assist drivers by providing real-time distance feedback to obstacles It is employed in smart parking solutions, utilizing ultrasonic sensors to assist drivers by providing real-time distance feedback to obstacles [19]. The Arduino Uno is integral in fire safety applications and It also automates railway gate operations.

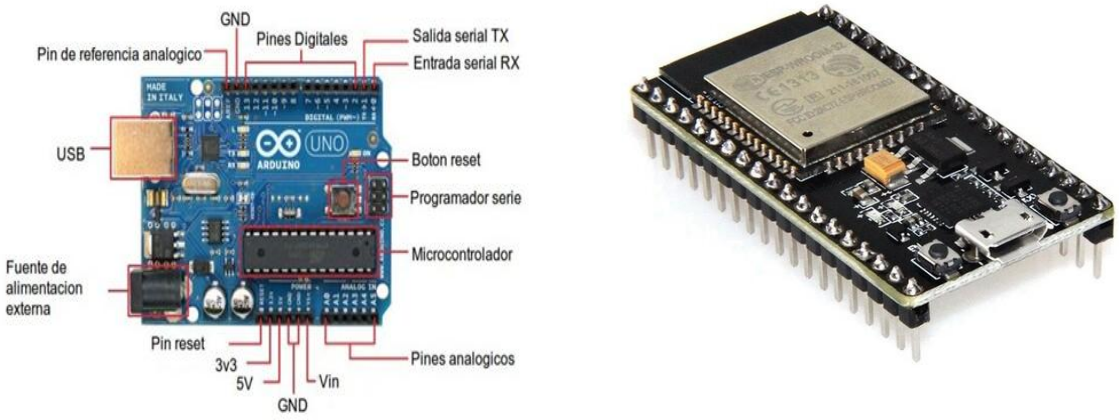


Fig.3.1: Arduino Uno and Esp32

3.1.2 ESP32

The ESP32 microcontroller is a versatile and powerful platform designed for Internet of Things (IoT) and embedded system projects. It is known for its low cost, low power consumption, and integrated Wi-Fi and Bluetooth capabilities, making it an ideal choice for a wide range of applications. The ESP32 is powered by a dual-core Tensilica Xtensa LX6 microprocessor, which provides robust processing power for various tasks. This microcontroller has been successfully implemented in numerous projects, showcasing its adaptability and efficiency in different domains. The ESP32 is utilized in smart door lock systems with facial recognition [20] and employed in non-intrusive presence detection systems using Wi-Fi [21].

3.1.3 Anemometer

Anemometers are essential instruments for measuring wind speed and direction, widely used in meteorology, wind energy, and various industrial applications. They come in different types, each with unique features and applications. The most common types include cup anemometers, which use rotating cups to gauge wind velocity, and vane anemometers, which combine a propeller with a directional vane. More advanced versions, such as ultrasonic and laser Doppler anemometers, use sound waves or laser technology for highly precise measurements without moving parts. Anemometers are essential for weather forecasting, environmental monitoring, and optimizing the performance of wind turbines. Despite their importance, they can be affected by factors like mechanical wear, dust, or extreme weather, making regular maintenance and calibration necessary for accurate results.

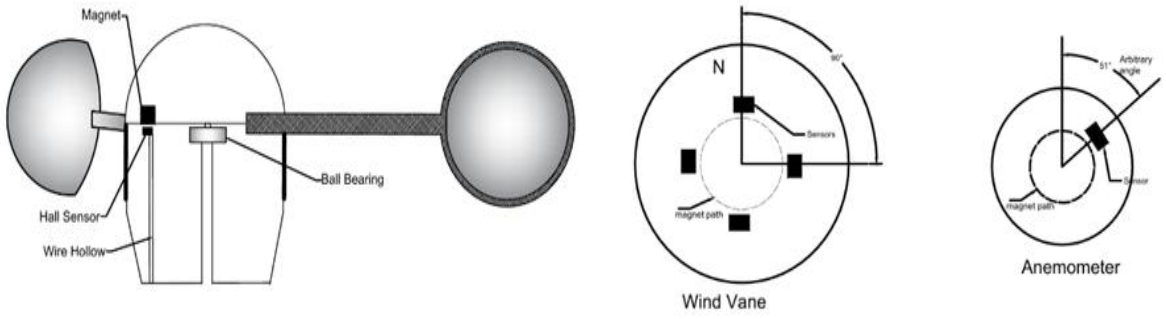


Fig.3.2: Anemometer and its sensor arrangement

3.1.4 Wind vane

Wind vanes are essential instruments for measuring wind direction, and recent innovations have enhanced their functionality and efficiency. These devices typically consist of a rotating vane mounted on a shaft, which aligns with the wind direction. Modern designs incorporate features such as wind power generation, electronic sensors, and advanced materials to improve performance and durability. These devices utilize sensors to provide precise measurements of wind direction and turbulence without mechanical contacts, improving reliability.Some wind vanes integrate wind power generators, utilizing fan blades to convert wind energy into electrical power, enhancing their utility. Wind vanes are capable of detecting light winds using thermoelectric sensors, which generate electrical outputs based on temperature differences caused by wind.

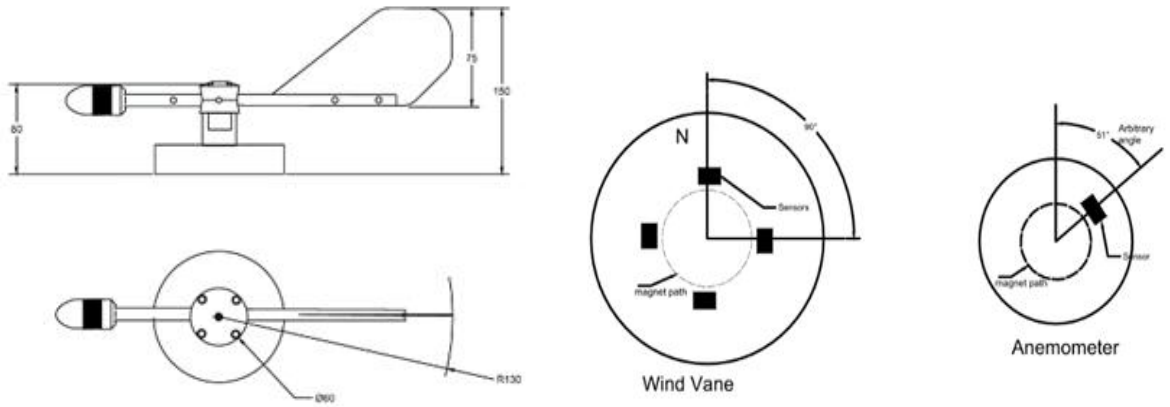


Fig.3.3: Wind Vane and sensor arrangement

3.1.5 Rain gauge

"Rain gauge" is a broad term, which can be used to refer to any instrument employed to measure the amount of liquid precipitation over a set period.Tipping Bucket rain gauge is a simple but very popular rainfall measurement sensor due to simple manufacturing structure consisting magnetic switching and counter. In TBRs water is collected by a funnel and channeled into a bucket which is part of two bucket vessel pivoted on a cylindrical axis. Once the bucket fills to a specified volume it tips raising the lower bucket into position. During this movement, a magnet in the tipping bucket mechanism generates a signal to an A3144 Hall effect sensor connected to Arduino. Each pulse corresponds to a nominal amount of rainfall based on the fixed Vt, adjustable by two calibration stop screws.

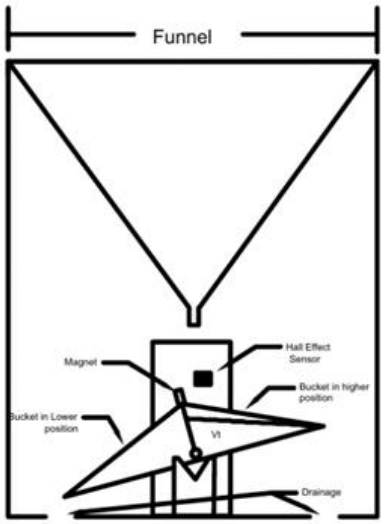


Fig.3.4: Tipping Bucket Rain Gauge

3.1.6 RTC Module

The Real-Time Clock (RTC) module is a crucial component in modern electronics, providing accurate timekeeping across various applications. RTC modules maintain precise time even during power outages, making them essential for devices like digital clocks, wearables, and automotive systems. They typically integrate a crystal oscillator and an RTC chip, often enhanced with temperature compensation to ensure accuracy under varying conditions. Recent advancements have led to ultra-low power RTC modules, achieving accuracies of ±1 ppm, which translates to minimal time deviation, crucial for applications requiring high precision [22]. Furthermore, these modules can include additional functionalities such as timers and alarms, enhancing their utility in energy-constrained environments.

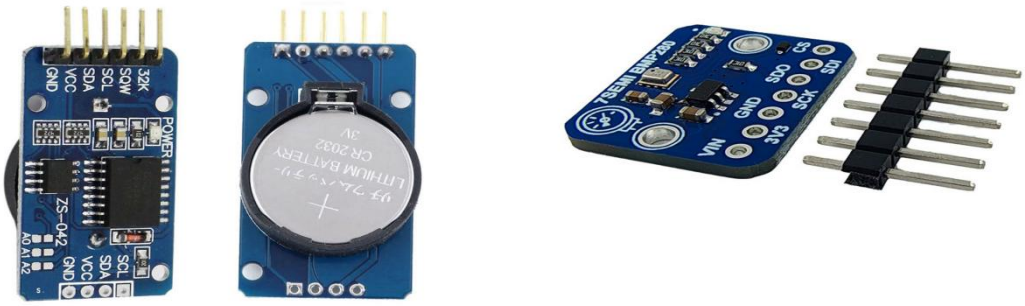


Fig.3.5: RTC Module & BMP280

3.1.7 BMP280

The **BMP280** is a high-precision environmental sensor developed by Bosch Sensortec, designed to measure **barometric pressure** and **temperature**. It is widely used in weather stations, GPS navigation, altitude tracking, and IoT-based environmental monitoring systems. The BMP280 is known for its compact size, low power consumption, and accurate readings, making it ideal for both mobile and embedded applications. It communicates using I²C or SPI interfaces, making integration with microcontrollers like Arduino and Raspberry Pi straightforward. The sensor can also indirectly calculate **altitude** based on pressure changes, which is useful for applications such as drones and wearables. With its high resolution and fast response time, the BMP280 is a reliable component for real-time atmospheric data collection.

3.1.8 DHT22

The DHT22 sensor is a widely used device for measuring temperature and humidity in various applications, particularly within the Internet of Things (IoT) domain. It is known for its accuracy and efficiency in monitoring environmental conditions, making it suitable for both industrial and personal use. The sensor is often integrated with microcontrollers like the Raspberry Pi to create real-time monitoring systems. These systems can be used in diverse settings, from human body temperature monitoring to environmental control in fermentation processes. The DHT22's ability to provide precise measurements, albeit with a slower response time, makes it a reliable choice for applications requiring consistent environmental data. The DHT22 sensor, when paired with a Raspberry Pi, can accurately measure temperature with a precision of up to 0.1°C, although it has a slower response to temperature changes compared to industrial-grade sensors [23]. It is also used in virtual simulations to monitor room temperature and humidity, demonstrating its capability to automate environmental control systems.

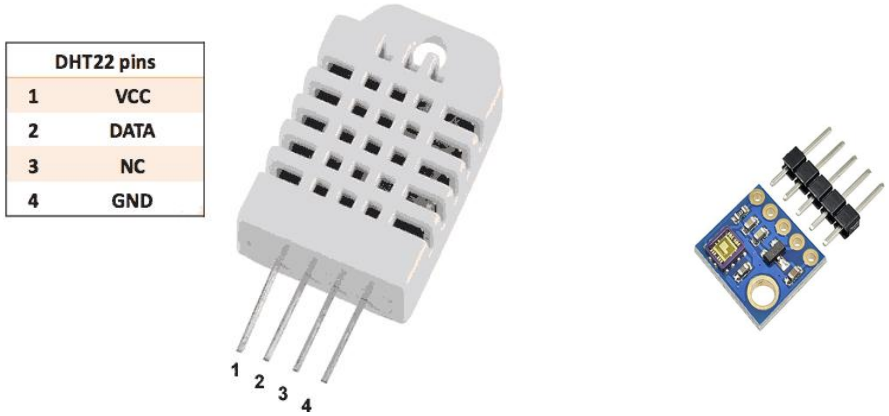


Fig.3.6: DHT22 & GYML8511

3.1.9 GYML8511

The GYML8511 is a UV sensor utilized in various applications, notably in detecting the authenticity of currency. This sensor operates by measuring the output voltage, which varies between genuine and counterfeit notes. For instance, research indicates that genuine Indonesian banknotes yield an output voltage of 1.01–1.02 V, while counterfeit notes printed on wax paper show higher voltages of 1.04–1.05 V. The GYML8511's integration with other components, such as the TCS3200 color sensor and audio feedback systems, enhances its functionality in automated detection devices. The GYML8511 effectively distinguishes between real and fake currency based on UV light reflection. It can be combined with other sensors and microcontrollers for enhanced performance in automated systems.

3.1.10 OV5647 Cmaera Module

The 5MP OV5647 Sensor is a high-quality, great-performance camera with robust build quality. The camera features an adjustable wide-angle 160° fisheye lens with night vision mode, capable of capturing 2592 × 1944 pixel static images, and supports 1080p @ 30 fps, 720p @ 60 fps, and 640 × 480 @ 60/90 fps video recording. It includes manual adjustable focus, and with the 160° field of view (FOV), it offers an exceptionally open vision—ideal for VR, AR applications, aerial photography, real-time car photography, indoor/outdoor monitoring, and general photo and video capture.

Features:

1. Angle of view: 64 × 48 degrees
2. Full-frame SLR lens equivalent: 35 mm
3. Fixed focus: 1 m to infinity
4. Max frame rate: 30 fps
5. Flex cable: 300 × 16 mm (neck width 6 mm)
6. Includes: 15 cm flat ribbon cable for 15-pin MIPI CSI interface.

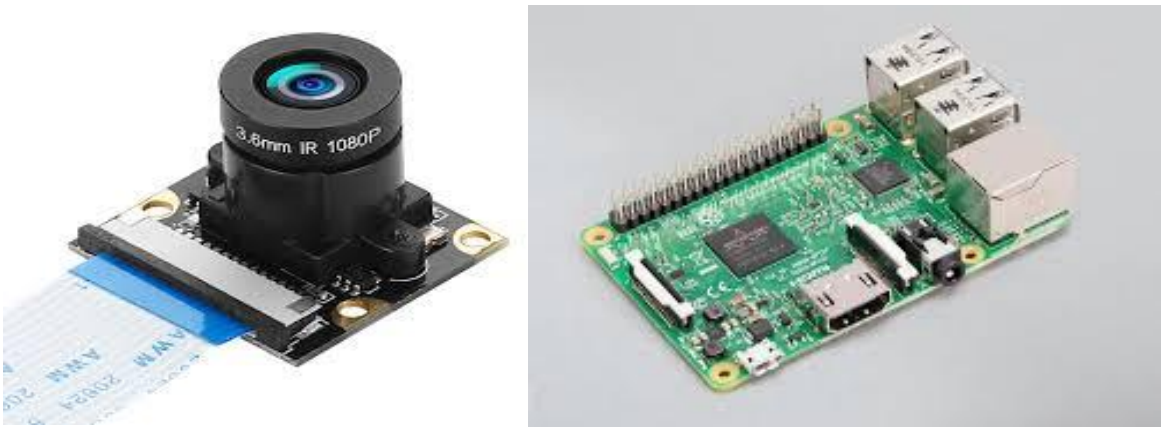


Fig.3.7: OV5647 Camera Module and Raspberry Pi 3 Model B+

3.1.11 Rasphberry Pi 3 Model B+

The Raspberry Pi 3 Model B+ is a compact and affordable single-board computer that offers improved performance and connectivity compared to its predecessors. Powered by a 1.4 GHz 64-bit quad-core ARM Cortex-A53 processor, it features 1 GB of LPDDR2 RAM, making it suitable for a wide range of applications, from DIY electronics projects to lightweight computing tasks. It comes with built-in Wi-Fi (802.11ac), Bluetooth 4.2, Gigabit Ethernet (over USB 2.0), and four USB 2.0 ports, allowing versatile connectivity. The board also includes a 40-pin GPIO header for hardware interfacing, CSI and DSI ports for camera and display modules, and HDMI output for connecting to monitors. Its low power consumption, broad community support, and compatibility with Raspbian (Raspberry Pi OS) make it an ideal platform for IoT, robotics, media centers, and educational use.

3.2 LSTM

3.2.1 Introduction of LSTM

Long Short-Term Memory Networks, commonly referred to as LSTM within the domain of deep learning, represent a specialized form of sequential neural network architecture. The LSTM model is known for its ability to remember significant events over long time sequences. This capability enables LSTM networks to excel in applications such as language modeling and time series forecasting, providing a significant advantage over conventional recurrent neural networks (RNNs). This advantage is particularly evident in fields like finance and healthcare,

where accurate predictions are crucial for decision-making [24]. Long Short-Term Memory (LSTM) networks integrate feedback mechanisms, thereby facilitating the analysis of entire sequences of data rather than merely isolated data points. This characteristic renders LSTM remarkably proficient in discerning and forecasting trends within sequential datasets such as time series, textual information, and auditory signals.

LSTM has become a powerful tool in artificial intelligence and deep learning, enabling breakthroughs in various fields by uncovering valuable insights from sequential data. The integration of LSTM in forecasting models has shown improved performance, particularly in short-term predictions, making it a preferred choice in many applications [14].
The effectiveness of LSTM networks in short-term forecasting is attributed to their advanced memory mechanisms, which allow them to capture long-term dependencies in data and improve predictive accuracy [25].

3.2.2 Fundamental concepts of the LSTM

Long Short-Term Memory (LSTM) networks have emerged as a powerful architecture within the realm of neural networks, particularly for sequential data. To understand LSTMs, it is essential to first explore foundational concepts, including an overview of neural networks and the specific class of Recurrent Neural Networks (RNNs) to which LSTMs belong.

- **Neural networks overview**

Neural networks are computational models inspired by the human brain's architecture and functionality. The architecture of a neural network typically consists of an input layer, one or more hidden layers, and an output layer. The connections between these layers are weighted, and the learning process involves adjusting these weights to decrease the difference between anticipated and actual outputs, typically via backpropagation [26]. They also consist of interconnected groups of nodes, or artificial neurons, which are organized into layers. Each neuron receives input, processes it through an activation function, and transfers the output to layers, as like human brain. Neural networks aim to identify patterns and make predictions based on input data. Recent advancements in neural networks have led to various architectures tailored to specific tasks, such as convolutional neural networks (CNNs) for image processing and recurrent neural networks (RNNs) for sequential data.

- **RNNs**

Recurrent Neural Networks (RNNs) represent a distinct category of neural network architectures specifically to handle sequential data. In contrast to conventional feedforward neural networks,

RNNs possess connections that recur upon themselves, thereby enabling them to retain a 'memory' of antecedent inputs.

Their architecture enables information retention across time steps, enhancing performance in tasks such as natural language processing and speech recognition using RNNs; however, they face challenges, including the vanishing gradient problem, which can hinder their ability to learn long-term dependencies effectively [27]. To mitigate these issues, advanced architectures like Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRUs) have been developed, enhancing the networks' memory capabilities.

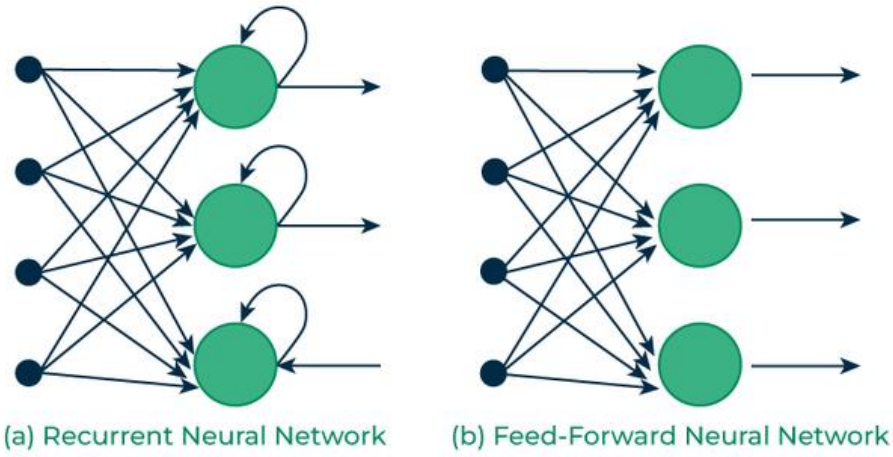


Fig.3.8: RNN Vs FFNN

The RNNs incorporate mechanisms to retain information over longer sequences, thereby improving the overall effectiveness of RNNs in various applications, including time series forecasting and sentiment analysis. The RNN architectures have led to significant improvements in predictive performance through these advancements, particularly in complex tasks that require understanding context over extended sequences.

3.2.3 Architecture of LSTM

LSTMs are particularly effective for tasks that involve sequential data, such as time series prediction, natural language processing, and speech recognition. The architecture of LSTM

networks is characterized by their unique gating mechanisms, which allow them to manage the flow of information across time steps effectively. This paper will elaborate on the three main gates in the LSTM architecture: the input gate, the forget gate, and the output gate.

- **Gates in LSTM**

The core innovation of LSTM networks lies in their use of gates, which are neural network components that regulate the flow of information into, out of, and within the cell state. Each

gate is governed by a sigmoid activation function, which outputs values between 0 and 1, determining how much information to let through. The gates work together to maintain and update the cell state, enabling the network to learn long-range dependencies in data.

- **Input gate**

The input gate controls the extent to which new information is added to the cell state. It consists of two main components: a **sigmoid layer** and a **tanh layer**. The sigmoid layer outputs values between 0 and 1, indicating which values in the input should be updated. The tanh layer generates candidate values that could be added to the cell state. The operations of the input gate can be mathematically expressed as follows:

Sigmoid Activation:

[it= σ(Wi.[ht-1, xt] + bi)] where it is the input gate's output Wi is the weight matrix, ht-1 is the previous hidden state, xt is the current input, and bi is the bias.

Candidate Values:

[Ct= tanh(Wc.[ht-1, xt] + bc)] where Ct represents the candidate cell state values.

The input gate determines which new data should be included in the cell state. It takes the current input (xt) along with the previous hidden state (ht−1), and processes them through a sigmoid layer that functions as a filter. Next, a tanh layer produces new candidate values, which are adjusted according to the output of the sigmoid layer. This procedure enables the LSTM to learn intricate patterns by carefully incorporating new information.

The final step involves combining these two outputs to update the cell state:

[Ct=Ct-1+it.Ct]

In this equation, Ct is the updated cell state, and Ct-1 is the previous cell state. The input gate thus effectively controls how much of the new information is allowed to influence the existing cell state.

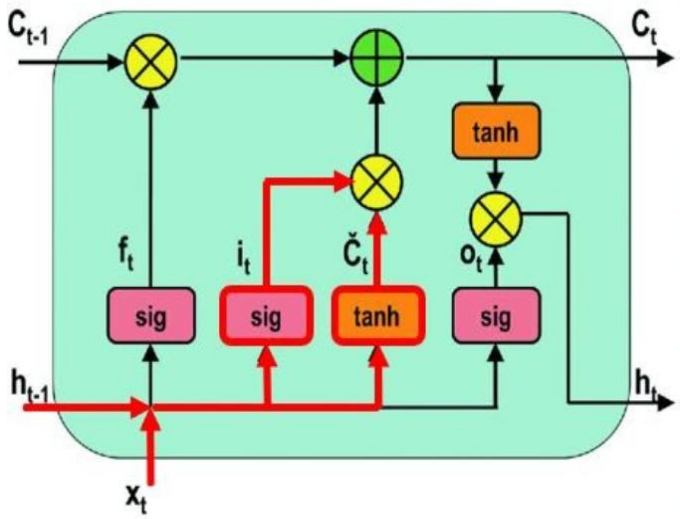


Fig3.9: Input Gate Operation of LSTM

- **Forget gate**

The forget gate determines what information from the previous cell state should be discarded. This gate is crucial for enabling the LSTM to forget irrelevant information, thereby maintaining a concise and relevant cell state. Similar to the input gate, the forget gate also uses a sigmoid activation function:

Forget Gate Activation:
[ft= σ(Wf.[ht-1, xt] + bf)]where ft is the output of the forget gate, Wf is the weight matrix for the forget gate, and bf is the bias.

The output of the forget gate multiplies the previous cell state, effectively "forgetting" certain aspects of it. It uses a sigmoid function to produce a number between 0 (completely forget) and 1 (completely keep) for each piece of information.

[Ct=ftCt-1]

In this equation, the forget gate output ft determines how much of the previous cell state Ct-1 is retained in the current cell state Ct. By controlling the retention of past information, the forget gate plays a vital role in the LSTM's ability to learn temporal dependencies.

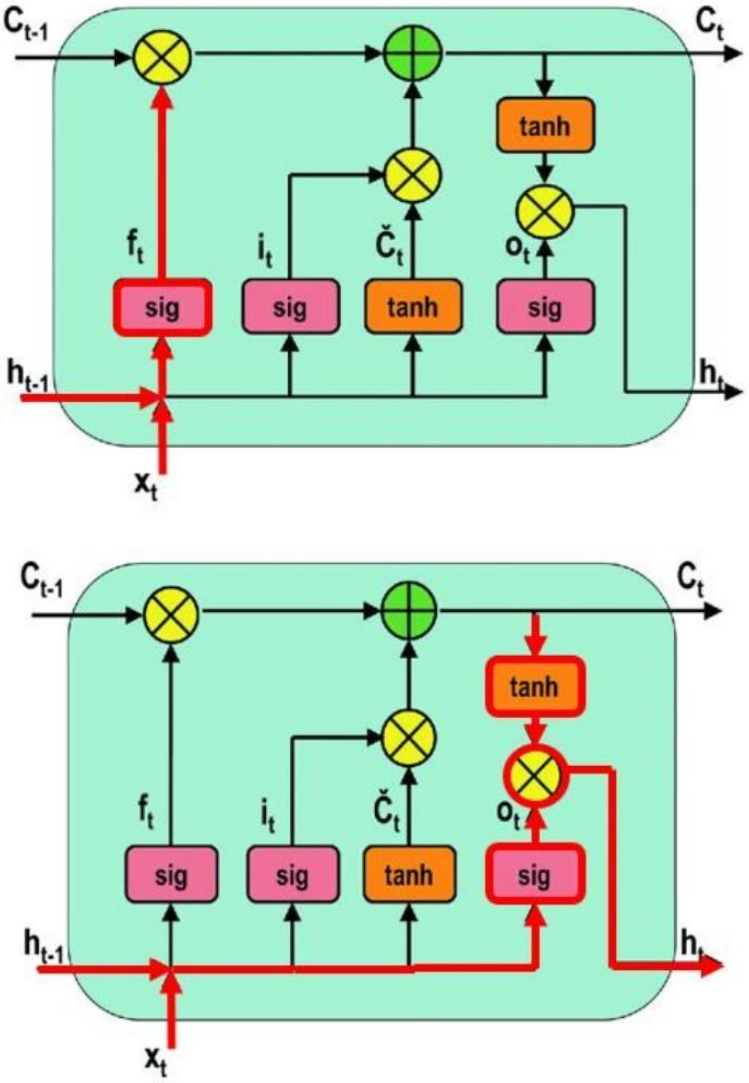


Fig3.10: Forget Gate and Output Gate Operation of LSTM

- **Output gate**

The output gate determines which information from the cell state should be output to the next hidden state. This gate, like the previous ones, utilizes a sigmoid activation function:

Output Gate Activation:
[ot= σ(Wo.[ht-1, xt] + b0)]where ot is the output of the output gate, Wo is the weight matrix for the output gate, and b0 is the bias.

The output gate uses the current cell state to produce the hidden state:

[ht=ot.tanhCt]

In this equation, ht the hidden state that will be passed to the next time step tanhCtensures that the cell state is squashed to a range between -1 and 1 before being output. The output gate thus facilitates the selective dissemination of information, allowing the LSTM to maintain relevant context while discarding unnecessary data.

3.2.4 Advantages of LSTM

Long Short-Term Memory (LSTM) networks provide a range of benefits that enhance their accuracy for numerous anticipation tasks, particularly in forecasting time series data. Their unique structure enables them to capture intricate dependencies within sequential information, rendering them more adept than conventional models in various applications. The primary benefits of LSTM networks are given below:

Enhanced Memory Capabilities

- LSTMs possess a memory cell that can maintain information over long periods, which is crucial for tasks requiring the retention of historical data
- This memory advantage allows LSTMs to outperform standard feedforward neural networks, particularly in scenarios with varying input sequence lengths.

Improved Forecasting Accuracy

- LSTMs excel in capturing nonlinear relationships in data, which significantly enhances forecasting accuracy compared to models like ARIMA [28].
- An ensemble approach using LSTMs can further improve prediction results by dynamically adjusting weights based on past errors, leading to better performance in time series forecasting.

Superior Performance with Large and Nonlinear Datasets

- LSTMs outperform traditional models like ARIMA when dealing with large datasets and nonlinear characteristics, providing better prediction accuracy in such scenarios

Computational Efficiency

- Recent advancements have led to methods that reduce training time while maintaining accuracy, such as combining LSTM with other techniques like Random Forest.
- The use of GPU acceleration in LSTM training allows for faster computations, facilitating both online and offline learning [29].

Handling Nonlinear Data

- LSTMs excel in capturing nonlinear statistical properties of time series data, which significantly improves forecasting accuracy. This capability is enhanced when LSTMs are used in ensemble methods that adaptively combine multiple models.

While LSTMs demonstrate significant advantages, they also come with increased complexity and longer training times compared to simpler models. This complexity can be a barrier for some applications, particularly where interpretability and speed are critical.

3.2.5 Application of LSTM

LSTM (Long Short-Term Memory) networks have emerged as a powerful tool across various domains, including natural language processing (NLP), time series prediction, speech recognition, and image captioning. Their ability to capture long-range dependencies makes them particularly effective in these applications. Below are key areas where LSTMs are applied:

Natural Language Processing (NLP)

- LSTMs excel in tasks such as language modeling and text generation due to their capacity to remember information over long sequences.
- They are often integrated into larger architectures, enhancing performance in tasks like sentiment analysis and machine translation.

Time Series Prediction

- LSTMs are widely used for forecasting due to their ability to model temporal dependencies in sequential data. Recent advancements, such as P-sLSTM, have improved their efficiency in long-term forecasting tasks [30].
- The Time Series Language Model (TSLM) has been developed to generate descriptive captions for time series data, enhancing interpretability and analysis.

Speech Recognition

- LSTMs are integral to speech recognition systems, where they process audio signals as sequences, effectively capturing phonetic and linguistic patterns.

Image Captioning

- LSTMs are employed in image captioning tasks, where they generate textual descriptions from visual inputs. Studies show that LSTM-based models outperform traditional RNNs and GRUs in generating accurate captions

Despite the impressive performance of LSTMs, challenges remain, particularly in managing extremely long sequences and the requirement for a substantial amount of training data in some cases. Other architectural models, such as transformers, are being investigated to address these challenges.

3.2.6 Challenges and Limitations

Long Short-Term Memory (LSTM) networks face several challenges and limitations that impact their performance and applicability. These issues stem from their complex architecture and inherent characteristics, which can hinder their efficiency and effectiveness in various contexts.

- Computational Complexity

Training Duration: LSTMs take longer to train compared to simpler models because of their intricate structure.

Resource Demands: They require more memory and processing power, particularly for lengthy sequences.

- Challenges in Parallelization

LSTMs sequentially handle sequences, which means that the output at each time step depends on the previous output. This characteristic restricts parallel processing and lengthens training time relative to models like Transformers.

- Vanishing and Exploding Gradients

While LSTMs manage this issue better than standard RNNs, they are not entirely free from the problems of vanishing or exploding gradients, particularly with very long sequences.

- Overfitting

LSTMs are prone to overfitting on smaller datasets due to their significant number of parameters, necessitating the use of regularization techniques like dropout.

- Sensitivity to Hyperparameters

LSTMs need precise tuning of parameters such as learning rate, number of layers, hidden units, and batch size. Suboptimal choices can lead to inadequate performance.

- Limitations in Long-Term Dependencies

Even though they perform better than standard RNNs, LSTMs can still find it challenging to manage very long-term dependencies, where crucial information is situated far back in the sequence.

- Interpretability of the Model

LSTM models operate as black boxes, making it challenging to understand or clarify how decisions are made or what the network has learned.

3.2.7 Comparison with other models

LSTM vs RNN

Long Short-Term Memory (LSTM) networks and Recurrent Neural Networks (RNNs) are both designed for sequential data processing, but they differ significantly in architecture and performance. LSTM networks are a specialized type of RNN that addresses the vanishing gradient problem, allowing them to capture long-term dependencies more effectively. This capability makes LSTMs particularly suitable for complex temporal patterns, as evidenced by their superior performance in various applications, including earthquake magnitude prediction and flood forecasting [31].

Key Differences in Architecture

- Memory Cells: LSTMs utilize memory cells and gates (input, output, forget) to manage information flow, enabling them to retain information over longer sequences.
- Gradient Management: RNNs often struggle with vanishing gradients, while LSTMs mitigate this issue, allowing for better learning of long-term dependency

Performance Comparison

- Accuracy: In studies, LSTMs generally outperform RNNs in tasks requiring long-term context, such as sentiment analysis and hydrological forecasting, achieving lower error metrics like RMSE.
- Simplicity vs. Complexity: RNNs may perform better on simpler datasets due to their less complex structure, as shown in certain flood forecasting scenarios.

Computational Cost

- LSTM is more computationally expensive due to additional gates and memory cells compared to RNNs.

Table 3.1: Difference between RNN and LSTM features

Features	RNN	LSTM
Design	Simple structure with a single repeating unit	Complex unit with gates (input, forget, output)
Memory Handling	Used for Short-term memory	Designed for long-term memory retention
Use Case	Simple time series, character-level tasks	NLP, speech recognition, long-sequence prediction

LSTM vs GRU

Among the various architectures that have been developed to address the limitations of traditional RNNs, Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRU) have gained significant attention due to their effectiveness in capturing long-range dependencies. Both LSTM and GRU are designed to mitigate the vanishing gradient problem, which hampers the training of standard RNNs by allowing gradients to either vanish or explode during backpropagation through time. Despite their shared goal, LSTM and GRU differ in their architectural components and operational mechanisms, leading to varied performance characteristics in different contexts. This paper elucidates the distinctions between LSTM and GRU networks and provides insights into their respective advantages and applications.

- **Architecture Complexity**:

  LSTM: The LSTM architecture consists of three gates—input, forget, and output gates—along with a memory cell that maintains information over long periods. This complexity allows LSTMs to manage information flow effectively, but also increases the number of parameters, which can lead to longer training times.

  GRU: The GRU simplifies this structure by combining the forget and input gates into a single update gate, along with a reset gate. This reduction in complexity results in fewer parameters, making GRUs generally faster to train and easier to optimize.

- **Gating Mechanisms**:

  LSTM: The three gates in LSTM networks serve distinct functions. The input gate controls the extent to which new information is added to the memory cell, the forget gate determines what information should be discarded, and the output gate regulates what information is sent to the next layer.

  GRU: The GRU uses an update gate to manage both the input and forget operations simultaneously, which means it can more fluidly update its memory based on the current input and the previous hidden state. The reset gate, on the other hand, determines how much of the past information to forget when computing the new hidden state.

- **Memory Cell**:

  LSTM: The LSTM's memory cell is a separate entity that allows for the explicit storage of information over time. This separation provides LSTMs with a robust mechanism for retaining and discarding information, which is particularly beneficial for tasks requiring long-term memory.

  GRU: In contrast, GRUs do not have a separate memory cell; instead, they directly compute the hidden state based on the current input and previous hidden state. This design choice can lead to more efficient memory usage and faster computation, but may limit the GRU's ability to capture very long-term dependencies compared to LSTMs.

- **Performance on Tasks**:

  LSTM: LSTMs have been shown to outperform GRUs in certain tasks that require learning long-term dependencies, such as language modeling and machine translation. Their three-gate structure allows for more nuanced control over information retention and processing.

  GRU: On the other hand, GRUs often perform comparably to LSTMs in many applications and may even surpass them in scenarios where the dataset is smaller or the task complexity is lower. Their simpler architecture can lead to faster convergence and reduced overfitting.

- **Training Time and Resource Efficiency**:

  LSTM: Due to their complexity, LSTMs typically require more computational resources and longer training times. This can be a significant consideration in large-scale applications where computational efficiency is paramount.

  GRU: GRUs, with their reduced number of parameters, generally require less training time and computational power, making them a more attractive option for applications where resources are limited or where rapid prototyping is necessary.

- **Suitability for Different Applications**:

  LSTM: Given their strength in handling long sequences and maintaining information over time, LSTMs are often preferred in applications such as video analysis, natural language processing, and any task requiring an understanding of context over extended periods.

  GRU: GRUs are frequently utilized in scenarios where speed is crucial, such as real-time applications or when working with smaller datasets. They have been effectively applied in areas like speech recognition and music generation, where the sequence length may not be excessively long.

- **Research and Community Adoption**:

  LSTM: LSTMs have been extensively studied and are widely adopted in the research community. Their established presence in literature means that there is a wealth of resources available for practitioners looking to implement LSTM-based models.

  GRU: While GRUs are gaining popularity, especially in recent years, they may not have the same level of established research backing as LSTMs. However, their growing acceptance in the machine learning community suggests a positive trend toward their utilization in various applications.

3.3 CNN

3.3.1 Introduction to Convolutional Neural Networks:

Convolutional Neural Networks (CNNs) represent a transformative advancement in the realm of artificial neural networks, particularly in the domains of image recognition and pattern analysis. CNNs are meticulously designed to handle grid-like data, such as images, thereby emulating the cognitive processes of human visual perception. These networks comprise a multitude of layers, including convolutional, pooling, and fully connected layers, each of which plays an integral role in the acquisition of intricate visual patterns. The convolutional layer, which serves as the backbone of a CNN, employs filters or kernels to traverse the input data, executing convolution operations to discern features such as edges, textures, and shapes across various levels of abstraction. As the data traverses through the network, CNNs construct a hierarchy of features, enabling the detection of complex structures that evolve from lower to higher orders. Pooling layers further augment this feature extraction process by diminishing the spatial dimensions of

feature maps, thereby reducing computational expenses while preserving essential information [32].

Recent advancements in Convolutional Neural Networks (CNNs) include the implementation of techniques like transfer learning and data augmentation, which enhance model performance while simultaneously reducing training time. This is accomplished through the use of pre-trained models and the synthetic enlargement of datasets. These advancements demonstrate the flexibility and capability of CNNs in tackling a wide array of tasks that go beyond simple image classification, encompassing areas such as video analysis, medical diagnosis, and natural language processing. The continuous development of CNN architectures and methodologies underscores their critical role in the progressive growth of artificial intelligence, especially in fields that necessitate high levels of accuracy in pattern recognition.

3.3.2 Basic Principles of CNNs

Convolutional Neural Networks (CNNs) are known for their capability to autonomously develop hierarchical representations of data from unprocessed input, which makes them particularly effective for handling tasks associated with structured grid-like data, like images. The fundamental concepts that form the basis of CNNs consist of:

1. Convolution Layers: The convolutional layer is a key component of Convolutional Neural Networks (CNNs) and plays a crucial role in the network's capability to extract and learn features from input data, particularly images. Unlike the traditional fully connected layers present in standard neural networks, convolutional layers apply a mathematical operation called convolution. This technique involves moving a filter or kernel across the input image to generate feature maps, allowing the network to recognize local patterns like edges, textures, and shapes. Notably, these patterns remain consistent despite translation, making convolutional layers effective at identifying objects in various spatial arrangements [32].

A significant feature of convolutional layers is their parameter-sharing approach, which greatly decreases the number of parameters in the network. By utilizing spatial hierarchies in the input data, convolutions enable the same weights to be applied across different sections of the input, thereby reducing the complexity of the overall design. This use of spatial relationships contributes to a more efficient learning process, making CNNs especially well-suited for extensive image analysis tasks. As the network advances through successive convolutional

layers, it captures increasingly sophisticated features, evolving from basic edges in the early layers to
detailed object representations in the deeper layers. This gradual refinement not only improves the model's ability to discriminate but also creates a solid framework for image classification, object detection, and a diverse range of computer vision applications.

In actual applications, convolutional layers are frequently paired with activation functions like Rectified Linear Unit (ReLU), which add non-linearity to the model, enhancing its ability to capture complex relationships within the data. Furthermore, pooling layers are typically employed after convolutional layers to diminish spatial dimensions and computational demands while maintaining crucial features, thereby further enhancing the effectiveness of CNN architectures. In summary, the convolutional layer emerges as an essential element of the CNN framework, supporting the deep learning techniques that have transformed the landscape of artificial intelligence.

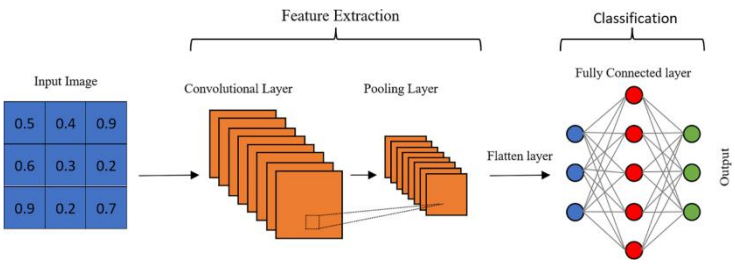


Fig.3.11: Layer in CNN

2. Activation Function: Activation functions play a crucial role in determining whether a neuron should be activated based on the inputs it receives in the network. These functions employ mathematical operations to assess the significance of the input for making predictions. If an input is considered significant, the function will "activate" the neuron. An activation function generates an output by processing a set of input values that are given to a node or layer. A node in a neural network functions similarly to a neuron in the human brain, as it accepts input signals (external stimuli) and responds to them. The brain evaluates input signals

and decides whether to activate a neuron based on established pathways and the intensity of the neuron's firing.

In deep learning, activation functions serve a comparable purpose. Their primary function is to convert the aggregated weighted input from a node into an output value that either gets passed to the subsequent hidden layer or is used as the final output. Various types of activation functions have been employed in this research, as illustrated in the below Figure. However, this discussion will specifically concentrate on the Rectified Linear Unit (ReLU). The ReLU function is currently the most commonly utilized activation function in neural networks. A significant benefit of ReLU in comparison to other activation functions is that it does not activate all neurons simultaneously. From the visual representation of the ReLU function above, it's noted that negative inputs are transformed into zero, leading to the neuron not being activated. This attribute makes it computationally efficient, as only a limited number of neurons are activated at any given moment. Additionally, it does not face saturation in the positive region. In practical applications, ReLU can converge six times faster than the tanh and sigmoid activation functions.

3. Pooling Layer: The pooling layer is an essential part of Convolutional Neural Networks (CNNs), mainly serving to unify the features obtained from the preceding convolutional layers. Pooling operations, known as subsampling or downsampling, are purposefully utilized following convolutional layers to efficiently decrease the dimensionality of the feature maps. This process reduces both computational load and parameter count while also lowering memory usage, which are crucial factors in deep learning applications. There are two primary categories of pooling:

Max pooling: In the Figure, while the filter slides over the input, it chooses the pixel with the highest value to transmit to the output array. In addition, this method is generally utilized more frequently contrasted with average pooling.

Average pooling: While the filter traverses the input, it computes the average value within the receptive field to transmit to the output array.

Although significant information is diminished in the pooling layer, it comes with several advantages to CNN. They assist in minimizing complexity, enhancing efficiency, and reducing the risk of overfitting.

4. Fully-connected Layer: Fully connected layers, often referred to as linear layers, link each input neuron to every output neuron and are widely utilized in neural networks. Figure 3.4

demonstrates an example of a compact fully connected layer that features four input neurons and eight output neurons. These layers are a crucial element of Convolutional Neural Networks

(CNNs), which have shown great success in recognizing and classifying images in the field of computer vision.

The CNN process starts with convolution and pooling, which breaks the image into features and examines them separately. The outcome of this process is then input into a fully connected neural network framework that determines the final classification decision.

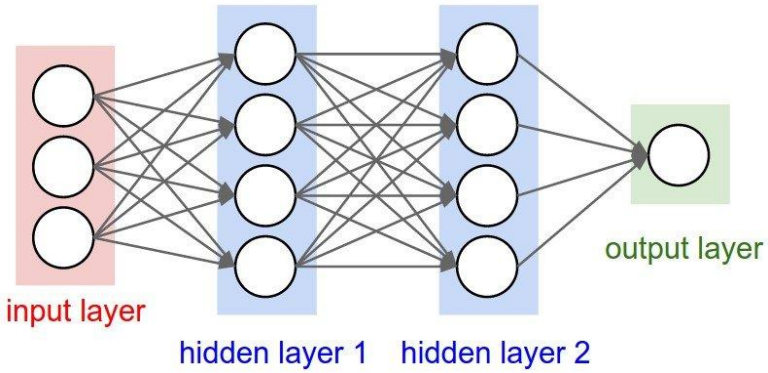


Fig3.12: Fully-connected layer with 3 input and 1 output neurons

3.3.3 Architecture of CNNs

The architecture of Convolutional Neural Networks (CNNs) is crucial for their effectiveness and capability to handle complex data. A standard CNN framework comprises a series of convolutional layers followed by pooling layers, ultimately leading to fully connected layers. CNN architectures are characterized by their depth, width, and connectivity patterns, which significantly influence their capacity to learn intricate patterns from data. Some notable CNN architectures are:

- **LeNet-5:** Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner created a convolutional neural network (CNN) based architecture called LeNet. The LeNet-5 architecture was developed to recognize handwritten and machine-printed characters. The 32 × 32 gas sensor feature matrix serves as the input layer. The convolutional layers C1 and C3 include 6 and 16 convolutional kernels. Following convolution, the outputs of C1 and C3 are 6 matrices measuring 28 × 28 and 16 matrices measuring 10 × 10, respectively. The pooling

layers S2 and S4 share a 14 × 14 kernel size. There are 120 neurons in the F5 layer and 84 in the F6 layer.

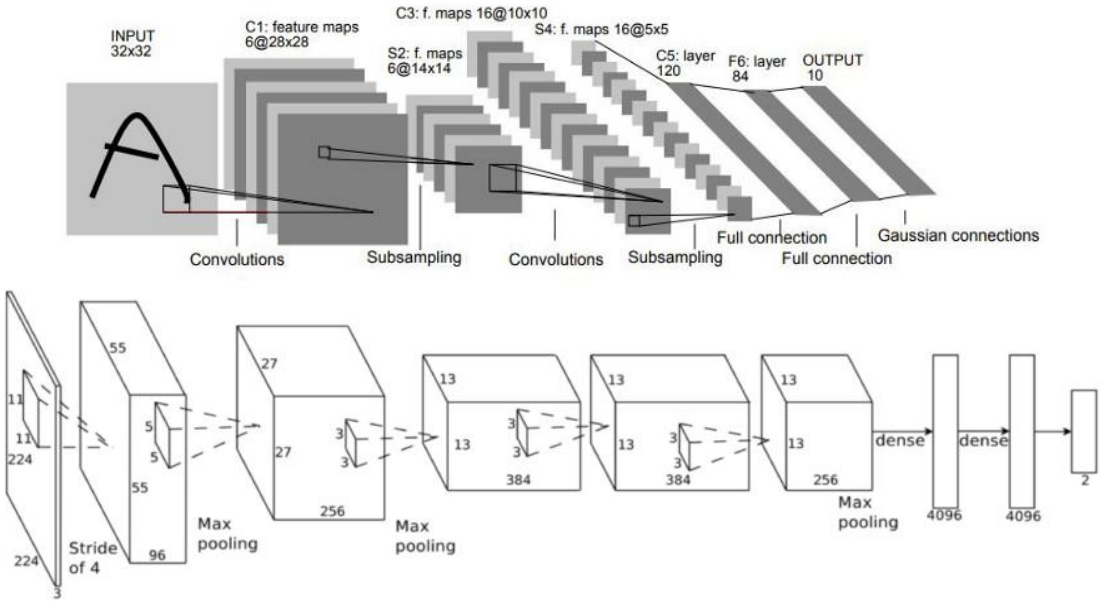


Fig3.13: LeNet-5 and AlexNet architecture

- **AlexNet:** Developed by Alex Krizhevsky et al., AlexNet gained widespread acclaim for its breakthrough performance in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2012. In the below Figure, it consists of eight layers, including five convolutional layers and three fully connected layers, along with the extensive use of dropout regularization and ReLU.

- **VGGNet:** VGGNet, developed by the Visual Geometry Group at the University of Oxford, is notable for its straightforward and consistent architecture. It consists of several convolutional layers utilizing small 3x3 filters, followed by max-pooling layers, and ultimately leads to fully connected layers for the purpose of classification. In this Figure, during the training process, the input image is maintained at a size of 224 × 224 pixels, while the receptive fields for all filters remain the same pixels across the entire network. This design enables the fully convolutional network to learn around 140 million parameters. Predictions are made using upsampling layers with four channels corresponding to all classes [n cl] in the reference dataset. The upsampling layers are combined with 1 × 1

convolutions from the third and fourth pooling layers sharing the same channel dimensions. The final upsampling layer generates precise details by integrating information from the last convolutional layer. The third and fourth pooling layers are upsampled with a stride of 8.

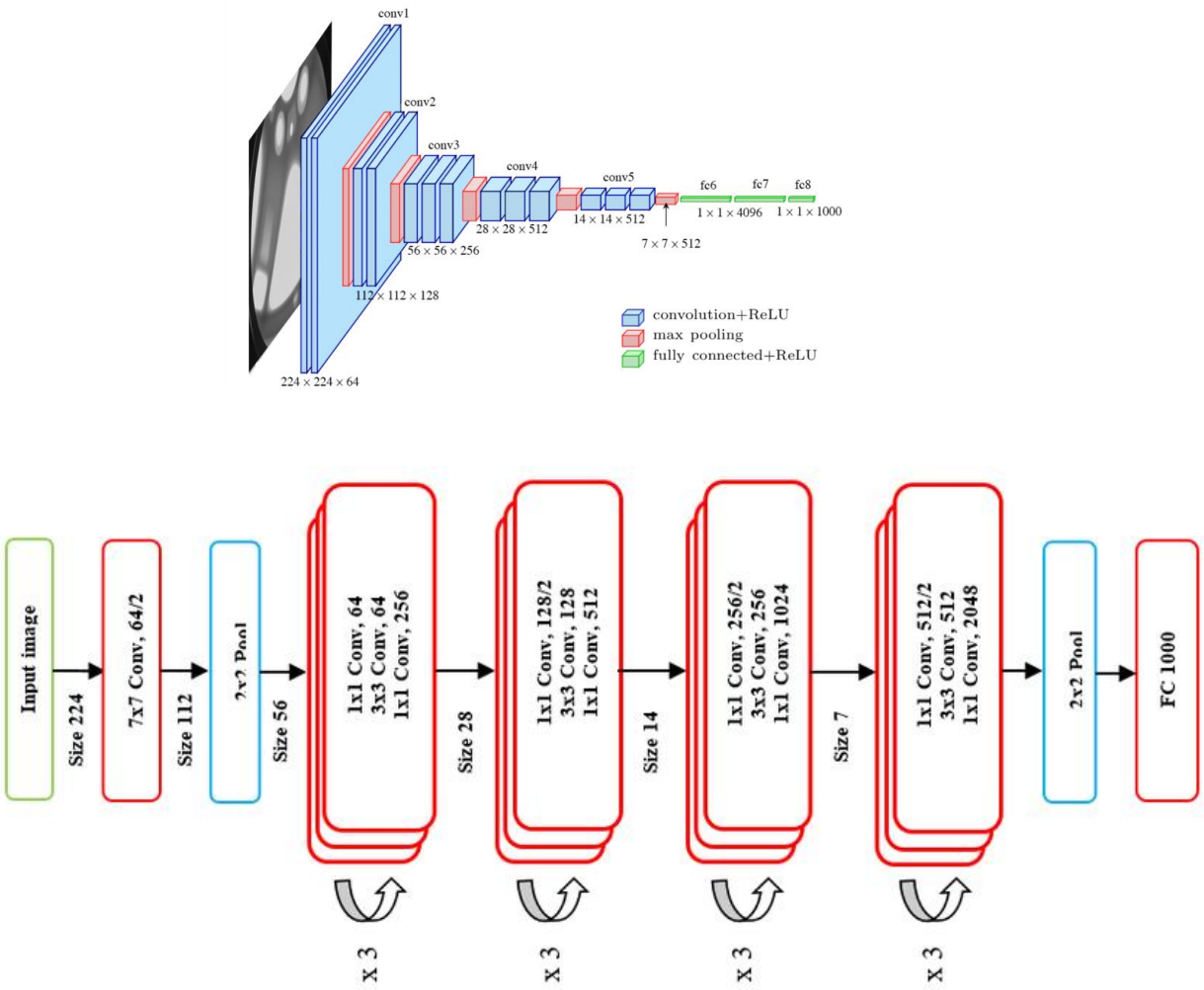


Fig.3.14: VGGNet and ResNet architecture

- **ResNet:** Residual Networks (ResNets) introduced the concept of residual learning, which incorporates skip connections to enhance the gradient flow during training. This architectural advancement allowed for the effective training of much deeper networks, resulting in enhanced performance on difficult tasks [33]. In the below figure ResNet includes a single convolution and pooling layer, succeeded by four layers with comparable functions.

3.3.4 Application of CNNs

Convolutional Neural Networks (CNNs) have become increasingly popular across various fields, demonstrating versatility and efficiency in multiple applications. CNNs are integral to some tasks such as image classification, object detection, and scene understanding, with applications in healthcare, autonomous vehicles, and surveillance [34].

Some key applications of CNNs are:

- **Image classification:** Image classification is a key area in artificial intelligence, utilizing Convolutional Neural Networks (CNNs) to identify objects in images. The progress in image classification methodologies has been significantly influenced by advancements in deep learning. The success of CNNs in classifying images depends on various factors, one of which is the careful selection of hyperparameters. Hyperparameters are essential configurations that determine the training process; they encompass aspects like learning rate, batch size, and the quantity of layers and filters in the network. Choosing the right hyperparameters is a complex task and greatly affects the model's performance on classification challenges. The empirical research conducted by Patel and Patel emphasizes the necessity of systematically examining these hyperparameters and their effects on CNN effectiveness, particularly when using datasets such as CIFAR-10. Their research findings highlight how precise tuning of hyperparameters can result in significant enhancements in classification accuracy, illustrating the need for continued exploration in this area.

- **Object detection:** Object detection is a key task in the field of computer vision that focuses on recognizing specific objects in images. The development of Convolutional Neural Networks (CNNs) has notably enhanced object detection capabilities since deep learning models like R-CNN were introduced, marking a significant advancement in this area. The R-CNN methodology begins with the selective search algorithm, which generates around 2,000 potential regions within an image.

- **Face recognition:** Facial recognition technology has become more recognized through the use of Convolutional Neural Networks (CNNs), which are highly effective in handling and examining visual data. CNNs, which are modeled after biological mechanisms, excel in

identifying patterns and features within images, making them ideal for facial recognition tasks. They operate by employing various convolutional layers that derive hierarchical features from facial images, which helps in pinpointing specific traits that differentiate one face from another. Recent advancements in this field indicate that CNNs can accurately classify and interpret facial expressions, which are essential for grasping human emotions. In a notable study, scientists developed a CNN that could identify human faces and categorize them into seven different emotional expressions, achieving an approximate accuracy rate of 60%. The performance of this system can be affected by several variables, including the quality and variety of the training dataset, optimization of network parameters, and the chosen network

architecture. Among the different architectures evaluated, VGG-16 was identified as the most effective, exhibiting improved performance in detecting facial expressions.

- **Medical image analysis:** The use of CNNs in the analysis of medical images has attracted considerable interest due to their capability to enhance and automate diagnostic procedures. In the field of dermatology, CNNs are essential for categorizing skin lesions, assisting healthcare providers in the early identification and precise diagnosis of various skin disorders, such as melanoma and other types of skin cancer. CNNs utilize extensive collections of labeled dermatological images to identify distinguishing features related to different skin conditions. By examining textural, structural, and contextual elements within the images, CNNs can efficiently distinguish between benign and malignant lesions, offering valuable assistance to dermatologists in their decision-making. However, the implementation of CNNs in the analysis of dermatological images presents several challenges, including the necessity for annotated datasets, the interpretability of the models, and the ability to generalize across various patient demographics and imaging scenarios.

3.3.5 Limitation of CNN

While Convolutional Neural Networks (CNNs) have demonstrated remarkable success in various field, they also exhibit certain limitations:

- **Large Dataset Requirement:** Convolutional Neural Networks (CNNs) depend significantly on extensive labeled datasets for successful training. A lack of adequate data can result in inadequate generalization and erroneous predictions.

- **Computational Intensity:** The design of CNNs typically incorporates a large number of parameters, resulting in significant computational requirements. The process of training deep CNN architectures can be lengthy and resource-heavy, which restricts their scalability in environments with limited resources.

- **Lack of Interpretability:** Convolutional Neural Networks (CNNs) are commonly called "black box" models due to the difficulty in interpreting their prediction processes. Understanding the decision-making process of CNNs is crucial, especially in medical applications where interpretability is essential for trust and acceptance.

- **Sensitivity to Adversarial Attacks:** Convolutional Neural Networks (CNNs) can be easily deceived by adversarial examples—minor alterations to input data that can lead to erroneous predictions.
- **Limited Contextual Understanding:** Although CNNs are highly effective at identifying local patterns and features in images, they can have difficulty comprehending wider contextual information. Grasping context is essential for tasks like scene interpretation and object identification in intricate settings.

- **Inability to Handle Variations:** Convolutional Neural Networks can have difficulty identifying objects that are rotated, resized, or shown from various angles, which can restrict their usefulness in practical situations.

Addressing these limitations requires ongoing research and development in the field of deep learning. Novel architectures, regularization techniques, interpretability methods, and robust training strategies can help mitigate the drawbacks of CNNs and improve their effectiveness in various fields

In summery, The theoretical analysis chapter explored both the computational and hardware foundations of this research. On the computational side, it introduced Convolutional Neural Networks (CNNs) for extracting spatial patterns from sky images and Long Short-Term Memory (LSTM) networks for capturing temporal dependencies in solar irradiance forecasting. Their integration was discussed as a hybrid framework capable of handling spatiotemporal data efficiently. On the hardware side, the chapter outlined essential components used in solar and

weather data collection, including Arduino and ESP32 microcontrollers for data acquisition and transmission, and various sensors such as the anemometer, rain gauge, wind vane, and DHT22 for measuring wind speed, rainfall, wind direction, temperature, and humidity, respectively.

Chapter 4: Dataset

4.1 Dataset Description

The primary dataset used in this research is the Stanford SKIPP'D dataset, which is designed specifically for deep-learning-based solar forecasting. The dataset consists of two levels:

- **Benchmark Dataset:** Contains three years of processed sky images (64×64 pixels) and concurrent PV power generation data recorded at a 1-minute interval. This dataset is ready for direct application in deep learning experiments.
- **Raw Dataset:** Provides high-resolution sky images (2048×2048 pixels) recorded at 20 frames per second, along with the corresponding PV output. This dataset is ideal for customized data extraction and alternative forecasting tasks.

The SKIPP'D dataset is supported by an extensive code base for data preprocessing and baseline model development. Research outputs using this dataset have shown that even down-sampled 1-minute sky image data can be instrumental in inferring solar panel output and forecasting power generation trends7. The dataset is publicly available through the Stanford repository, making it an ideal benchmark for academic exploration and pilot studies in solar forecasting.

4.2 Dataset Generation

A custom-built IoT-based system was developed for real-time weather data acquisition. The s system integrated multiple sensors and modules, including Arduino Uno, ESP32, Anemometer, BMP280 (barometric pressure), DHT22 (temperature and humidity), Rain Gauge, Wind Vane, RTC module (for time and date stamping), and a fishing camera for sky image capture. Each component played a specific role in measuring key environmental parameters such as wind speed, wind direction, atmospheric pressure, humidity, temperature, and rainfall.

4.2.1 Weather Parameters dataset

The developed weather station is a microcontroller-based embedded system designed for real-time environmental monitoring. It consists of two primary components: an outdoor sensing unit and an indoor ground unit, as illustratedThe outdoor sensing unit, built around an Arduino Uno, integrates and manages multiple atmospheric sensors. These sensors are responsible for measuring seven essential weather parameters, including, Ambient temperature, Relative humidity, Wind direction, Wind speed, Barometric pressure, Rainfall accumulation, Ultraviolet (UV) light intensity

The Arduino Uno collects data from all sensors at regular intervals and transmits the readings via wired serial communication (UART) to the indoor unit.The indoor ground unit is powered by an ESP32 microcontroller with built-in Wi-Fi capability. It serves as a communication gateway that receives data from the Arduino Uno, processes it, and uploads it to the cloud. The ESP32 ensures reliable data handling and utilizes Google Sheets as a lightweight and accessible cloud storage platform for real-time remote monitoring.

To store the data in Google Sheets, the system employs a Google Apps Script Web App as a cloud endpoint. The ESP32 sends hourly sensor data in HTTP POST requests formatted as JSON or URL-encoded key-value pairs. The script parses the incoming data and appends it to a designated Google Sheet. Each row in the sheet typically includes:Timestamp (real-time clock-based), Temperature (°C) ,Humidity (%), Wind Speed (m/s), Wind Direction (°) ,Barometric Pressure (hPa), Rainfall (mm), UV Index.

This method provides an easy-to-access, cost-effective, and platform-independent solution for cloud-based environmental data logging, without requiring dedicated servers or proprietary platforms.The system is configured to log sensor readings every hour. These readings are timestamped and stored individually in Google Sheets, allowing long-term trend analysis and historical review. In future iterations, additional functionality such as daily summary computation (e.g.max/min/average) and graphical dashboard integration may be included.

	A	B	C	D	E	F	G	H
1	Timestamp	Wind Speed (km/h)	Wind Direction	Temperature	Humidity (%)	Rainfall (mm)	UV Intensity (mV	Pressure (hPa)
2	2025-01-10 00:00:00	2.2	N	57.3	96.27	0	0	1018
3	2025-01-10 01:00:00	0.9	N	60	94.93	0	0	1018
4	2025-01-10 02:00:00	0.9	N	57.3	96.16	0	0	1018
5	2025-01-10 03:00:00	2.2	NE	56	95.05	0	0	1018.5
6	2025-01-10 04:00:00	0.7	NW	58.5	96.15	0	0	1018
7	2025-01-10 05:00:00	1.3	N	55.3	95.51	0	0	1018
8	2025-01-10 06:00:00	2.2	NE	55.1	93.48	0	0	1018.9
9	2025-01-10 07:00:00	1.3	N	55.1	93.06	0	0	1020
10	2025-01-10 08:00:00	1.3	NE	56.4	89.04	0	1	1021
11	2025-01-10 09:00:00	0.1	N	60	76.05	0	3	1021.2
12	2025-01-10 10:00:00	1.6	SE	62.8	86.3	0	5	1021
13	2025-01-10 11:00:00	2.2	E	62	69.92	0	6	1020
14	2025-01-10 12:00:00	0.2	E	69.9	74.01	0	7	1019.1
15	2025-01-10 13:00:00	2.9	SE	69.3	63.91	0	7	1017
16	2025-01-10 14:00:00	2.9	SE	73.5	55.57	0	6	1017
17	2025-01-10 15:00:00	4.6	W	76.9	57.09	0	5	1016
18	2025-01-10 16:00:00	2	S	77.1	73.12	0	3	1016
19	2025-01-10 17:00:00	2	W	69.9	82.87	0	1	1016
20	2025-01-10 18:00:00	0	W	67.9	85.65	0	0	1015.9
21	2025-01-10 19:00:00	4.5	N	68.4	86.52	0	0	1017
22	2025-01-10 20:00:00	3.8	N	66.1	90.88	0	0	1017
23	2025-01-10 21:00:00	2.2	N	62.3	89.17	0	0	1017.8
24	2025-01-10 22:00:00	3.1	N	61.8	88.54	0	0	1018
25	2025-01-10 23:00:00	2.9	N	61.8	89.62	0	0	1018
26	2025-01-11 00:00:00	2.2	N	60	93.1	0	0	1018.3
27	2025-01-11 01:00:00	2.9	N	59.2	89.55	0	0	1018
28	2025-01-11 02:00:00	2.5	N	58.4	91.27	0	0	1018
29	2025-01-11 03:00:00	0	N	57.1	95.35	0	0	1018.5
30	2025-01-11 04:00:00	1.3	N	47	93.01	0	0	1018
31	2025-01-11 05:00:00	1.6	NE	54.6	94.24	0	0	1018

Fig.4.2: Hardware Setup for data Collection

4.2.1 Sky Image Dataset

To capture continuous sky observations for the purpose of solar irradiance forecasting, a full video dataset was generated using the OV5647 camera module interfaced with a Raspberry Pi 3 Model B+ at Sylhet Engineering College, Bangladesh. The OV5647 is a 5-megapixel CMOS image sensor capable of high-resolution video capture, making it well-suited for monitoring dynamic sky conditions such as cloud movement and formation. The Raspberry Pi platform was selected due to its compact design, extensive GPIO support, and seamless integration with camera modules and automation tools.

```
city = LocationInfo("Sylhet", "Bangladesh", "Asia/Dhaka", 24.8949, 91.8687)
storage_base = "/mnt/external/sky_videos"

def is_daytime():
    s = sun(city.observer, date=datetime.date.today(), tzinfo=city.timezone)
    now = datetime.datetime.now(city.timezone)
    return s["sunrise"] <= now <= s["sunset"]

def record_video():
    now = datetime.datetime.now()
    date_folder = os.path.join(storage_base, now.strftime("%Y-%m-%d"))
    os.makedirs(date_folder, exist_ok=True)
    filename = f"sky_{now.strftime('%H%M%S')}.h264"
    filepath = os.path.join(date_folder, filename)

    subprocess.run([
        "libcamera-vid",
        "-t", "60000",           # 60 seconds
        "-o", filepath
    ])

while True:
    if is_daytime():
        print("Daytime detected. Recording video...")
        record_video()
    else:
        print("Not daytime. Sleeping for 15 minutes...")
        time.sleep(900)
```

Fig.4.3: OpenCV Commands for capturing data

The camera setup was installed in an outdoor location within the campus premises to provide an unobstructed hemispherical view of the sky. A fisheye lens was used to ensure a wide field of view, enabling effective coverage of cloud formations across the sky dome. Video acquisition was automated using Python scripts in conjunction with the `libcamera-vid` utility, which is part of the modern Raspberry Pi OS camera stack. To ensure the relevance of captured data to solar energy forecasting, recordings were restricted to daylight hours only. This was achieved by dynamically calculating local sunrise and sunset times using the Astral Python library based on the geographical coordinates of Sylhet.

Considering the large volume of video data generated daily, an external HDD storage device was connected to the Raspberry Pi via USB 3.0 (with a powered hub if required). The Pi was configured to automatically mount the external drive at boot using `fstab`, and all video recordings were directed to this external storage. This ensured uninterrupted long-term recording without overloading the Raspberry Pi's microSD card, which is typically limited in both speed and lifespan. File management scripts were also employed to organize videos by date and time and to periodically convert `.h264` recordings into image sequences using **FFmpeg**, with frames extracted at regular intervals (e.g., every 1 or 5 seconds).

These sky images shown in figure.4.4 serve as the primary input for the deep learning models developed for short-term solar nowcasting and forecasting. The dataset reflects real-time atmospheric conditions over Sylhet and provides valuable training data for spatiotemporal prediction tasks.

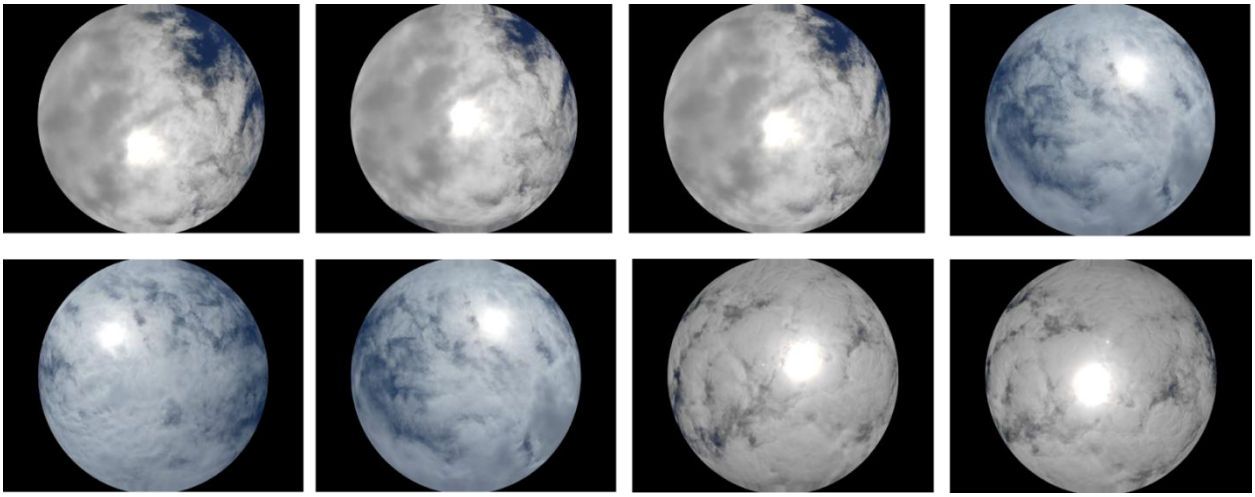


Fig.4.4: Example of collected sky image data

In summery, This research utilized both publicly available and custom-generated datasets for solar forecasting. The primary source was the Stanford SKIPP'D dataset, which includes benchmark sky images and PV output data at different resolutions and sampling rates, providing a reliable foundation for deep-learning experiments. In addition, a custom IoT-based weather station was developed to collect real-time environmental data using Arduino Uno, ESP32, and various sensors (anemometer, rain gauge, wind vane, BMP280, DHT22, UV sensor, and RTC). The system logged hourly readings of temperature, humidity, wind speed, wind direction, pressure, rainfall, and UV index, storing them in Google Sheets via ESP32 cloud integration. To complement these measurements, a sky image dataset was generated at Sylhet Engineering College using a Raspberry Pi 3B+ with an OV5647 fisheye camera module. Continuous daytime sky videos were recorded, processed with Python and OpenCV, and stored on external HDDs, with frames extracted for model training

Chapter 5: Methodology

5.1 Overview

This research paper implements two complementary models to forecast solar power generation i.e nowcast model and forecast model.

5.2 Nowcast Model Architecture

The nowcast model is a convolutional neural network (CNN) designed to extract relevant spatiotemporal features from sky images for the purpose of short-term solar power prediction. The architecture comprises a sequence of convolutional and normalization layers, followed by fully connected layers, each carefully structured to ensure effective feature learning and generalization.

- **Input layer:** Receives a sky image of size 64×64 with three color channels.
- **Initial Convolution Block:**The model begins with an initial convolutional block, which receives input in the form of RGB images with three channels. This block applies a 2D convolutional layer that projects the input to 32 output channels using a kernel size of 7 and padding of 3 to preserve the spatial dimensions. Following the convolution, batch normalization is applied to stabilize the learning process by normalizing the feature maps. A ReLU (Rectified Linear Unit) activation function introduces non-linearity, enabling the model to learn complex patterns. To prevent overfitting, dropout is applied, randomly deactivating a fraction of the neurons during training..
- **Second Convolution Block (Conv2):**The second block continues the feature extraction process by applying a 2D convolutional layer with 64 input channels and 128 output channels. A kernel size of 3 and padding of 1 are used to maintain spatial resolution while focusing on finer local features. This convolution is followed by batch normalization to enhance training stability, a ReLU activation function to introduce non-linearity, and dropout for regularization.
- **Third Convolution Block (Conv3):**In the third block, the network deepens the feature representation through a 2D convolutional layer that maps 128 input channels to 256 output channels. A kernel size of 3 and padding of 1 are used, similar to the previous block. Unlike the earlier layers, this block uses layer normalization instead of batch normalization.

- **Flatten and Fully Connected layers:** After the convolutional stages, the model includes a fully connected block to aggregate and transform the learned spatial features into a compact feature vector suitable for prediction. This block applies layer normalization to further stabilize the input to the final layers and includes dropout to reduce the risk of overfitting.
- **Output Layer:** Finally, the model produces its nowcast prediction through an output layer. Although the exact form of this layer is not explicitly detailed in the diagram, it typically consists of a linear regression layer that maps the high-level features to a scalar or sequence of scalar values representing the predicted solar power output

This architecture exploits the localized characteristics of the sky image, making it robust enough to estimate the current operating conditions of solar panels as observed in real-time weather conditions

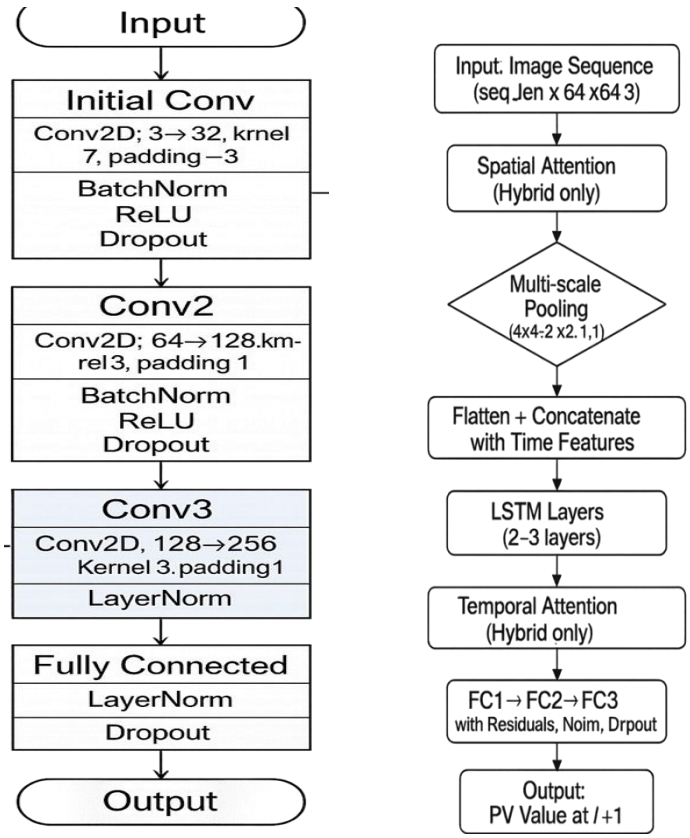


Fig.5.1. Nowcast and Forecast Model Architecture

\

5.3. Forecast Model Architecture

The forecast model is designed to predict near-future PV output by using a sequence of historical sky images and past PV power data. This problem is inherently spatiotemporal, as both the visual sky features and their evolution over time contribute to future solar generation.

The model employs a hybrid architecture that integrates CNNs and LSTM layers:

- **Input: Image Sequence (seq_len × 64 × 64 × 3):**The model begins by receiving a sequence of RGB images, where each image has a spatial resolution of 64×64 pixels and 3 color channels. The sequence length (seq_len) typically spans a temporal window (e.g., 10–30 minutes) to capture evolving cloud patterns relevant to solar irradiance prediction. This input serves as the temporal context for the forecasting model.
- **Spatial Attention (Hybrid only):** In the hybrid version of the model, a spatial attention mechanism is applied to focus on the most informative spatial regions across the input frames. This module computes attention maps to weigh different parts of the image more heavily, allowing the network to learn which spatial areas are most relevant for predicting solar output. It is particularly useful in highlighting dynamic cloud formations that affect PV generation.
- **Multi-scale Pooling (4×4, 2×2, 1×1):** To capture spatial features at different resolutions, multi-scale pooling is employed. Three pooling operations with varying kernel sizes (4×4, 2×2, and 1×1) are applied in parallel or sequentially. This helps the model extract both coarse and fine-grained features, improving its robustness to scale variations in cloud structures. The pooled features are concatenated to enrich the representation.
- **Flatten + Concatenate with Time Features:** The multi-scale spatial features are flattened into 1D vectors and concatenated with auxiliary time-related metadata, such as normalized hour of day, day of the week, and month. These temporal features help the model incorporate periodic patterns in solar generation. This concatenated vector forms a comprehensive spatio-temporal representation of the current scene.
- **LSTM Layers (2–3 layers):** The concatenated features are passed through a stack of 2 to 3 Long Short-Term Memory (LSTM) layers. These recurrent layers are capable of modeling

long-term dependencies in time-series data, enabling the model to learn from the sequential nature of the input images. The LSTM layers help in tracking the evolution of cloud movements and lighting conditions over time.

- **Temporal Attention (Hybrid only):** In the hybrid variant, a temporal attention module is added after the LSTM stack. This mechanism allows the model to assign different levels of importance to each timestep in the sequence, effectively letting it focus on the most critical moments that influence the forecast. It improves interpretability and performance by selectively attending to significant temporal contexts.
- **FC1 → FC2 → FC3 with Residuals, Norm, Dropout:** The output from the LSTM (or attention block) is fed into a series of fully connected (dense) layers. These layers refine the learned features and reduce dimensionality. Residual connections are used between layers to ease gradient flow and prevent vanishing gradients. Normalization (e.g., LayerNorm or BatchNorm) stabilizes training, while dropout regularizes the network by preventing overfitting.
- **Output: PV Value at t / t+1:**The final layer outputs a single value (or a small vector), representing the predicted photovoltaic (PV) power at the current or next timestep (t or t+1). This output can be interpreted as the short-term solar generation forecast and is used in grid management or energy storage decisions.

5.4. Implementation Details

5.4.1 Data Preprocessing and Integration

Preparing the SKIPP'D dataset for model training involves several critical steps. The dataset comprises both benchmark and raw data, necessitating preprocessing to conform to the model's input requirements:

- **Image Resizing:**

High-resolution images (2048×2048 pixels) are down-sampled to 64×64 pixels for efficient processing without significant loss of critical visual features7.

- **Normalization:**

Both sky images and PV generation values are normalized to have a consistent scale. Image pixel values are scaled to the range [0, 1] while PV outputs are standardized using z-score normalization.

- **Data Augmentation:**

Techniques such as random rotation, flipping, and cropping are applied to augment the training dataset, thereby reducing overfitting and enhancing model robustness.

- **Temporal Alignment:**

For the forecast model, it is essential to align the sequence of sky images with the corresponding PV measurements. This involves matching timestamps and potentially interpolating data when necessary, given that the PV data is recorded at 1-minute intervals7.

The data preprocessing pipeline is implemented using Python 3.6 and TensorFlow 2.x, following the guidelines provided in the open-source reference codes from the Stanford dataset repository.

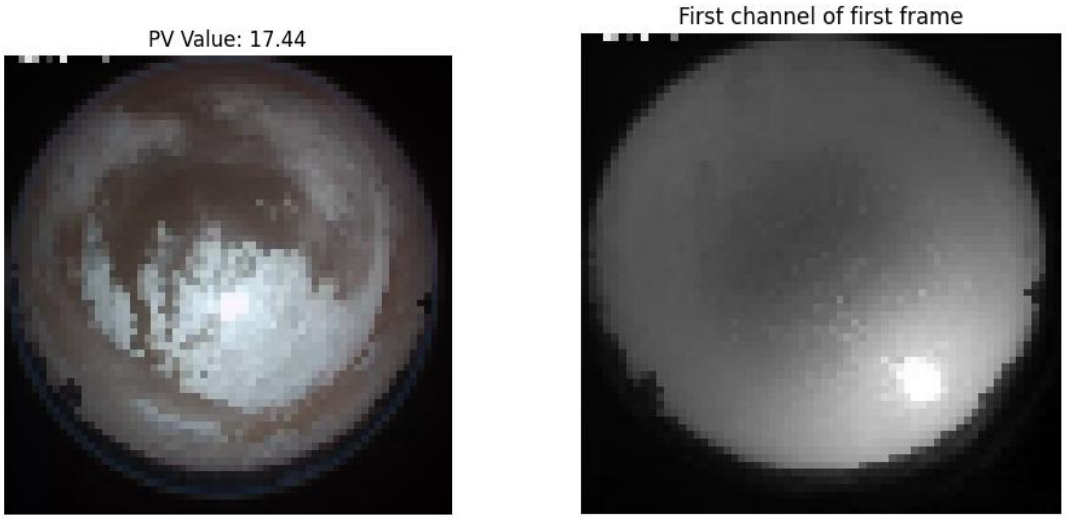


Fig 5.2: Visualization of a specific channel from the first frame of first image

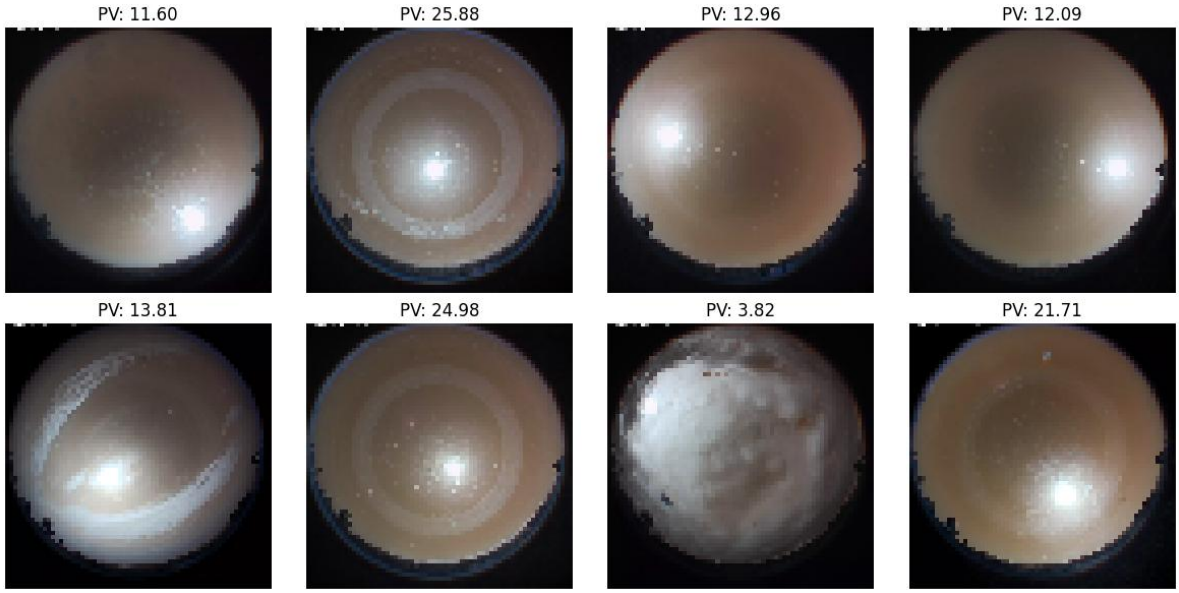


Fig.5.3: Samples of collected sky image data

5.4.2. Model Training and Regularization

➢ **Loss Function**

- **MSE (Mean Squared Error)** is used to penalize large deviations between predicted and true PV values. It is particularly effective for regression tasks where large errors need to be heavily penalized.

➢ **Optimizer**

- **Adam** is selected due to its adaptive learning rate and momentum-based update rules, offering efficient convergence for both shallow and deep networks. A fixed initial learning rate of **0.001** is used for stability and consistency across experiments.

➢ **Regularization**

- **Dropout** with a rate of **0.3** is applied during training to reduce overfitting by randomly deactivating neurons in the fully connected layers.
- **L2 Regularization** (weight decay) helps prevent the model from learning overly complex weight distributions, thereby promoting better generalization on unseen data.

➢ **Scheduler**

- **ReduceLROnPlateau** dynamically reduces the learning rate if the validation loss stagnates for a certain number of epochs, helping the optimizer fine-tune the model around local minima.

➢ **Training Parameters**

- A batch size of 32 balances convergence speed and memory usage.
- 50 training epochs allow sufficient time for the model to converge while leveraging early stopping (if used) to avoid overfitting.

Table 5.4.2: Training Configuration

Component	Nowcast	Forecast
Loss Function	MSE	MSE
Optimizer	Adam (lr = 0.001)	Adam (lr = 0.001)
Regularization	Dropout (0.3), L2	Dropout, L2
Scheduler	ReduceLROnPlateau	ReduceLROnPlateau
Batch Size	32	32
Epochs	50	50

The models are trained using TensorFlow 2.x on GPU NVIDIA RTX 3050 to achieve efficient and scalable training. Model hyperparameters (e.g., learning rate, batch size, number of epochs) are tuned using validation sets extracted from the benchmark dataset.

In summery, This research employs two complementary deep learning architectures for short-term solar forecasting: a Nowcast model and a Forecast model. The Nowcast model is a CNN-based network that processes individual sky images to extract spatial features and estimate current PV output conditions. In contrast, the Forecast model integrates CNNs with LSTM layers to analyze sequences of sky images and PV data, capturing both spatial and temporal dependencies for predicting near-future solar generation. The hybrid variant further incorporates spatial and temporal attention mechanisms along with multi-scale pooling to improve robustness. For data preparation, the Stanford SKIPP'D dataset is preprocessed through image resizing, normalization, augmentation, and temporal alignment of images with PV measurements.

Chapter 6: Evaluation Metrics and Results Analysis

Forecast accuracy is evaluated using a suite of established metrics. These metrics include:

- Mean Absolute Error (MAE): Measures the average magnitude of prediction errors without considering their direction and provides an interpretable measure in the same units as the PV output3.

$$MAE = (1/n) \times \Sigma |yi - \hat{y}i| \quad (1)$$

- Mean Squared Error (MSE): Provides a quadratic penalty to larger errors, thereby highlighting significant deviations in model predictions.

$$MSE = (1/n) \times \Sigma (yi - \hat{y}i)^2 \quad (2)$$

- Root Mean Squared Error (RMSE): Derives from the MSE and brings the error measure back to the original scale, facilitating practical comparisons3.

$$RMSE = \sqrt{[\,(1/n) \times \Sigma (yi - \hat{y}i)^2\,]} \quad (3)$$

- Mean Absolute Percentage Error (MAPE): Expresses forecast errors as a percentage, assisting in the comparison across different scales3.

$$R^2 = 1 - [\,\Sigma (yi - \hat{y}i)^2 / \Sigma (yi - \bar{y})^2\,] \quad (4)$$

The Nowcast hybrid model, which combines CNN and LSTM architectures, demonstrated strong predictive performance with a Mean Squared Error (MSE) of 6.078, indicating low average squared deviation from the true PV values. The Root Mean Squared Error (RMSE) was calculated at 2.465, reflecting the model's ability to limit large prediction errors. Additionally, it achieved a Mean Absolute Error (MAE) of 1.340, signifying high accuracy in capturing the average magnitude of errors without over-penalizing outliers. The R-squared (R^2) score of 0.896 suggests that the model explains a substantial portion of the variance in the actual power output, highlighting its effectiveness for short-term solar power estimation.

Table 6.1: Model Evaluation Values

	Model	MSE	RMSE	MAE	R-Squared
Nowcast	Hybrid (CNN+LSTM)	6.078461	2.465453	1.340176	0.895964
Forecast	Hybrid (ConvoLSTM +LSTM)	7.404557	7.404557	1.452039	0.873191

The Forecast hybrid model, built using a ConvLSTM followed by an LSTM stack, also exhibited robust performance for future PV prediction. It attained a Mean Squared Error (MSE) of 7.405, with a corresponding RMSE of approximately 2.721, indicating stable generalization over the forecast horizon. The model's MAE stood at 1.452, showcasing its reliability in maintaining a low average absolute deviation. An R^2 value of 0.873 confirms that the forecast model retains a strong correlation with ground truth values, making it suitable for anticipating solar power trends over extended lead times

6.1 Nowcast Model Evaluation

- The time series plot reveals a significantly tighter alignment between actual and predicted values, particularly prominent in the first 700 data points. This close match illustrates the model's strength in short-term prediction scenarios.
- The residual plot demonstrates that the majority of errors are tightly centered around zero, with a narrower distribution range. This reflects reduced occurrence of large deviations and contributes to the model's overall robustness.
- The predictions vs. actuals plot displays a denser clustering of data points along the diagonal line, indicating better accuracy and less dispersion compared to the Forecast Model.

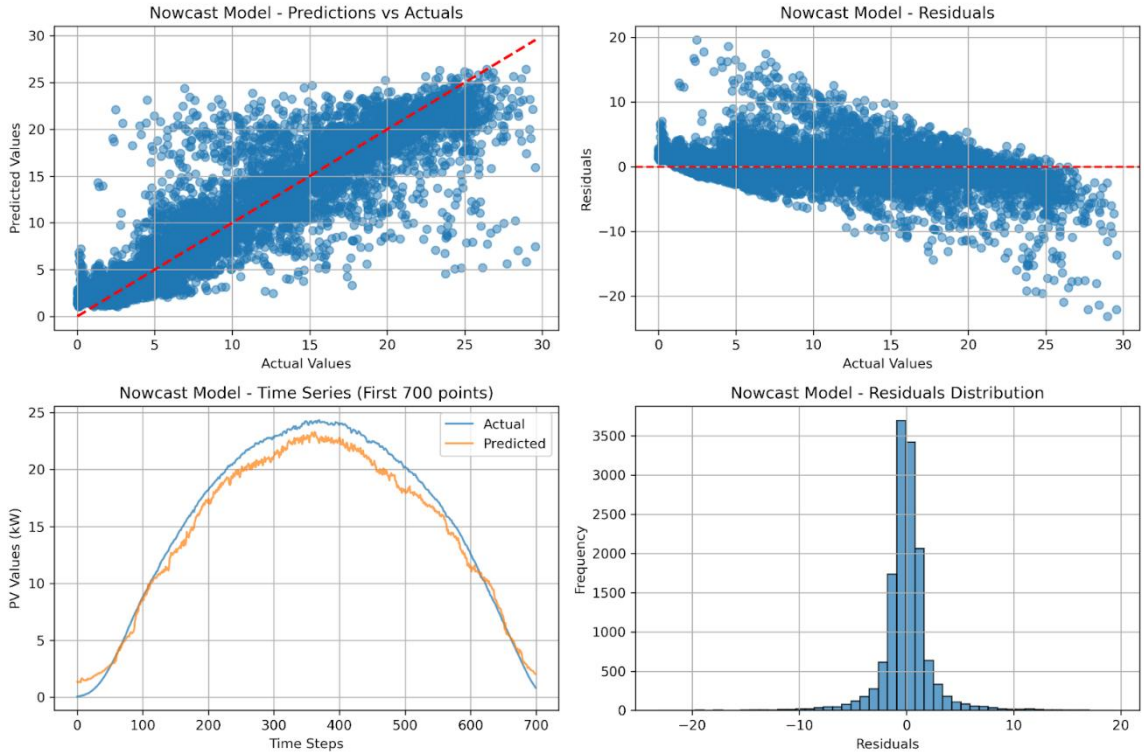


Fig.6.1: Nowcast Model test results

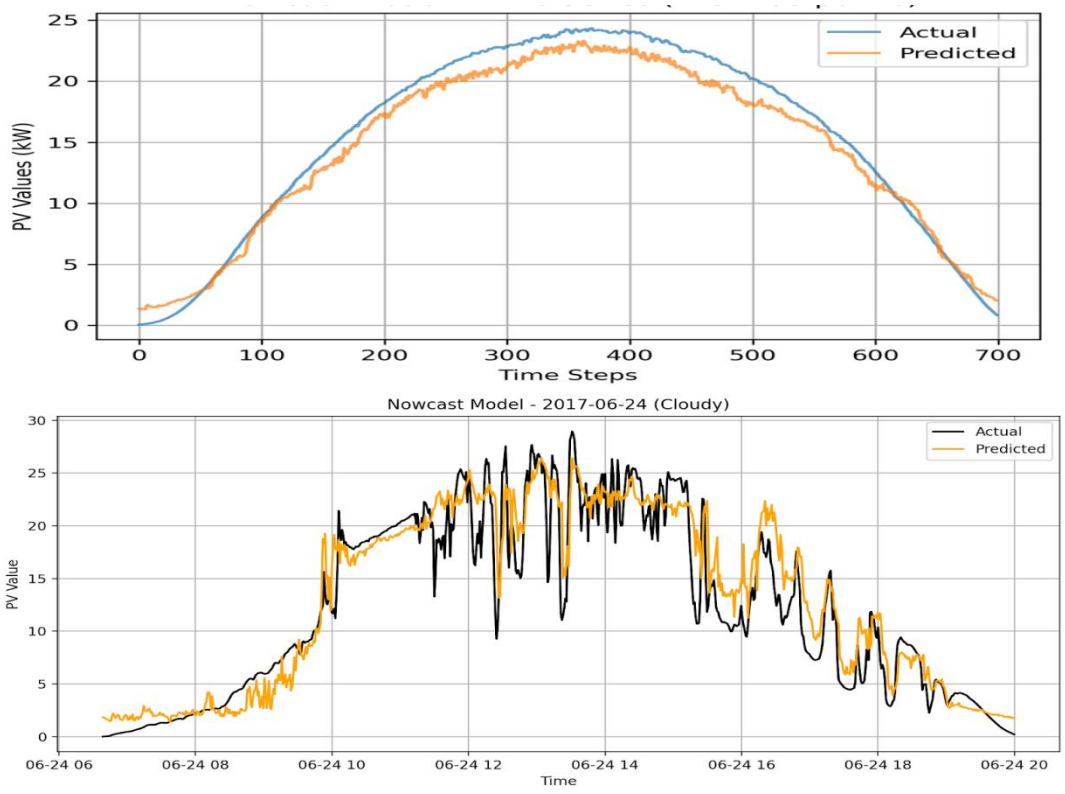


Fig.6.2: Nowcast Actual VS Predicted PV Values for a Sunny day and Cloudy day

6.2 Forecast Model Evaluation

- The time series plot of actual versus predicted photovoltaic (PV) power values shows that the model is generally able to follow the temporal trends. However, noticeable deviations are observed at certain time steps, indicating over- or under-prediction in specific intervals.
- The residuals scatter plot and histogram reveal a wider spread of error values, suggesting higher variance and more frequent occurrence of large prediction errors.
- In the predictions vs. actuals plot, while a positive correlation is evident, there is greater dispersion of points around the ideal diagonal line, implying reduced precision in prediction.

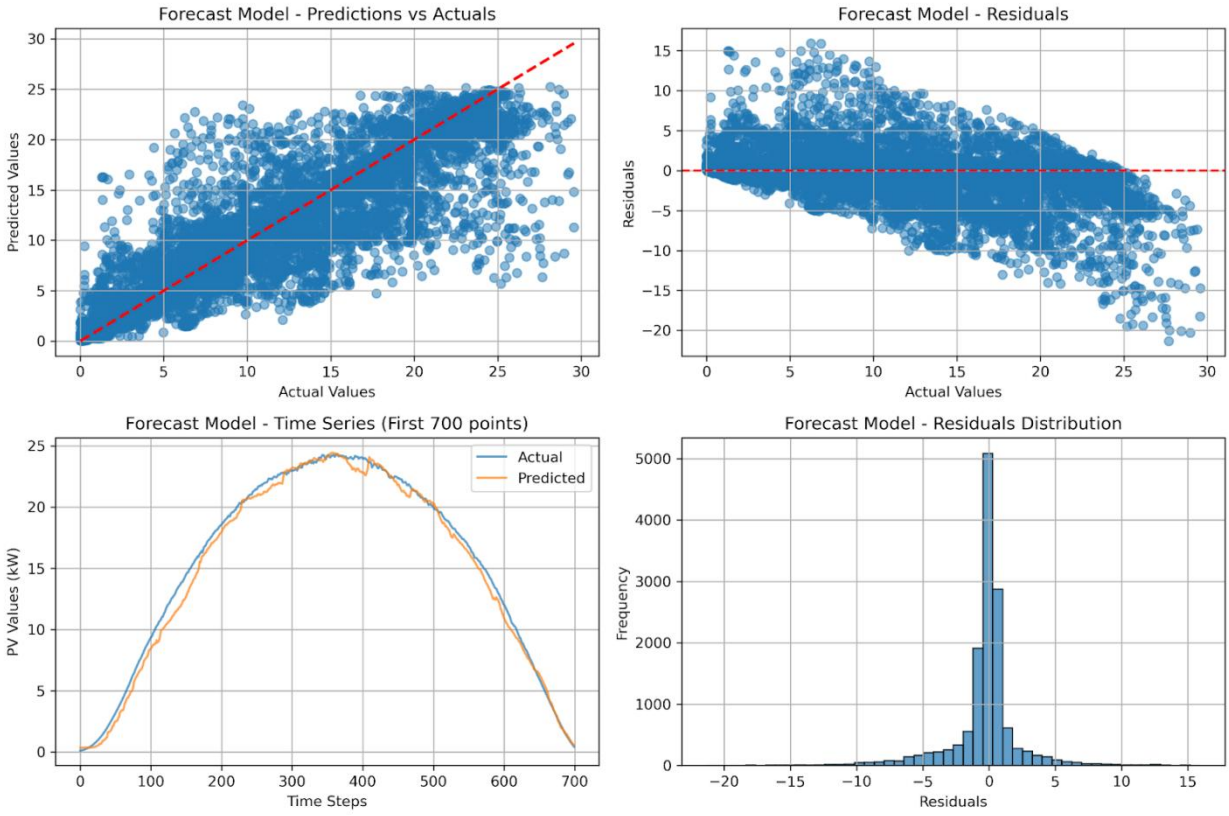


Fig.6.3: Forecast Model test results

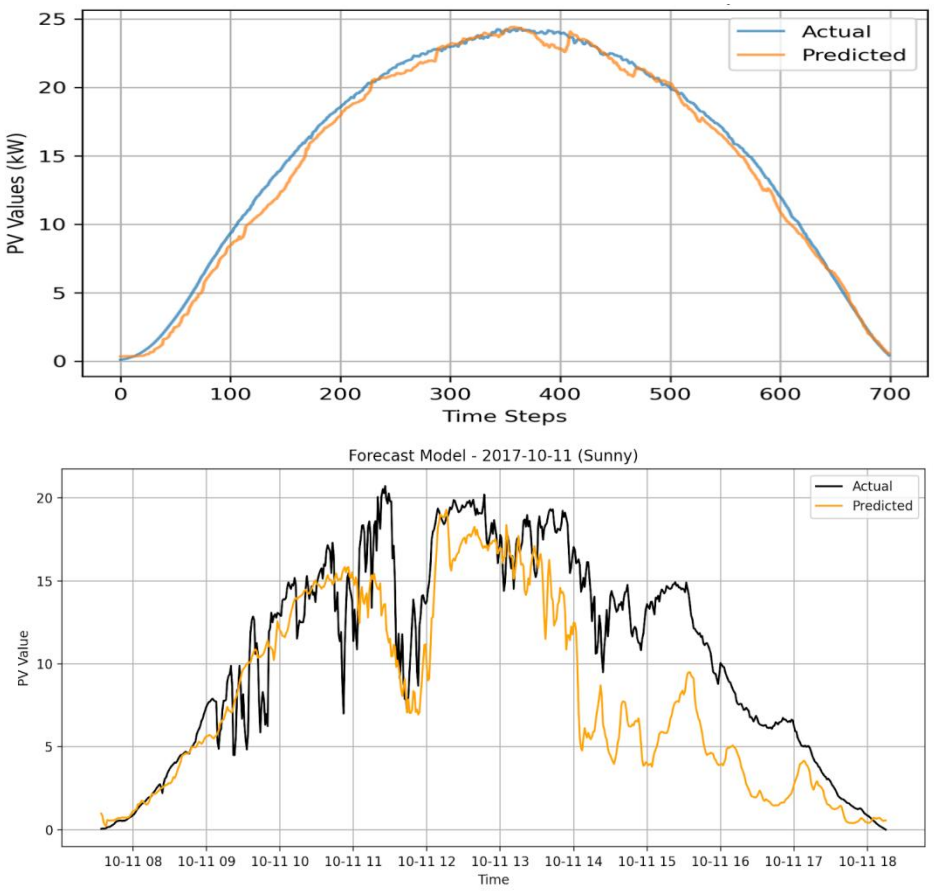


Fig.6.4: Forecast Actual VS Predicted PV Values for a Sunny day and Cloudy day

6.3 Discussion

The performance metrics clearly indicate that the proposed Nowcast hybrid model (CNN + LSTM) significantly outperforms the model reported in existing literature. Specifically, the Nowcast model achieves a Mean Squared Error (MSE) of 6.078. This lower MSE reflects reduced error variance, and the improvement is further emphasized by a Root Mean Squared Error (RMSE) of 2.465, substantially better than the 3.377 reported previously. The Mean Absolute Error (MAE) is also lower at 1.340 versus 2.410, indicating that the proposed model consistently makes smaller errors. Most notably, the R-squared (R^2) value reaches 0.896, showing that nearly 90% of the variability in PV output is explained by the model. These gains demonstrate that incorporating spatial features via CNN and modeling temporal sequences with LSTM leads to more accurate and generalizable nowcasting.

Table 6.3: Comparisn with existing works

	Model	MSE	RMSE	MAE	R-Squared
Nowcast	Hybrid (CNN-LSTM)	5.872247	2.423272	1.407359	0.899493
Forecast	Hybrid (ConvLSTM-LSTM)	7.404557	2.721132	1.452039	0.873191
SUNSET Mode [6]	Nowcast (CNN)	x	2.43	1.50	x
	Forecast(CNN)	x	3.03	1.71	x

Similarly, the proposed Forecast hybrid model (ConvLSTM + LSTM) shows superior performance over the literature baseline. It yields an MSE of 7.405, and an RMSE of approximately 2.721—a marked improvement over the 3.296 previously reported. The MAE of 1.452 outperforms the baseline 2.212, affirming its reliability in maintaining low prediction deviations. Furthermore, an R^2 score of 0.873 indicates better explanatory power. The use of ConvLSTM allows the model to directly capture spatio-temporal patterns in the input image sequence, which, when combined with LSTM for sequential forecasting, leads to better long-range performance.

In summery, Chapter 6 evaluates the predictive performance of the proposed hybrid models using standard metrics, including MAE, MSE, RMSE, MAPE, and R^2. The Nowcast model (CNN + LSTM) demonstrates strong short-term prediction capability, achieving an MSE of 6.078, RMSE of 2.465, MAE of 1.340, and R^2 of 0.896, reflecting high accuracy and tight alignment between predicted and actual PV values. Residual and prediction plots indicate minimal dispersion and robust performance under varying weather conditions.

The Forecast model (ConvLSTM + LSTM), designed for near-future PV prediction, attains an MSE of 7.405, RMSE of 2.721, MAE of 1.452, and R^2 of 0.873.

Chapter 7: Conclusion and Future Work

This study demonstrates the effectiveness of using sky image data combined with advanced deep learning models—specifically, Hybrid CNN-LSTM and ConvLSTM-LSTM architectures—for short-term solar forecasting. By leveraging both open-source datasets and real-time weather data collected through custom IoT devices, the research provides a comprehensive approach to predicting solar radiation and, consequently, solar power generation.

The integration of image-based data with numerical weather information enables the models to capture complex atmospheric patterns, particularly cloud movement, which is critical for accurate short-term predictions. The results indicate that deep learning approaches utilizing sky images outperform traditional numerical methods in terms of forecasting accuracy, as reflected by lower error metrics such as MSE, MAE, and RMSE, and higher R^2 values.

The CNN-LSTM Nowcast models MSE is 5.87, MAE is 1.40 is closer to ideal, RMSE is also closer 2.42 outperforms previous works. R^2 value is .899 is almost perfect prediction. The forecast ConvLSTM-LSTM model MSE is 7.40, MAE is 1.45 is closer to ideal. The RMSE is 2.72 outperforming previous forecast work. The R^2 value is 0.87 is near perfect score 1.

Ultimately, the findings highlight the potential of combining machine learning, IoT technologies, and image processing to enhance the reliability of solar energy forecasting. This advancement supports more efficient grid management, reduces reliance on fossil fuel-based backup systems, and contributes to the broader adoption of renewable energy by improving operational planning and cost management for solar power plants. The research underscores the importance of continued innovation in data collection and modeling techniques to further improve the precision and applicability of solar forecasting solutions.

Future Work

Looking ahead, there are several promising directions to further enhance the effectiveness and generalizability of solar power forecasting systems. Firstly, expanding the dataset by integrating additional parameters can significantly improve forecasting performance. While current models primarily rely on sky images and solar irradiance values, incorporating other meteorological and environmental variables—such as temperature, humidity, wind speed,

aerosol optical depth, and cloud type—can offer a more holistic view of the atmospheric conditions influencing solar generation. Such multimodal data fusion would allow the models to learn richer patterns and dependencies, ultimately resulting in more accurate and reliable predictions.

Secondly, the integration of diverse and advanced sensors presents an important step toward improving the quality and resolution of observational data. Instruments like pyranometers (for measuring global solar radiation), pyrheliometers (for direct beam solar irradiance), and high-resolution fisheye cameras (to capture the entire sky dome) can provide critical measurements and perspectives that are otherwise unattainable with standard camera setups. These sensors can help distinguish between diffuse and direct solar components and detect subtle cloud formations, both of which are essential for high-fidelity nowcasting and short-term forecasting. Enhanced sensor fusion would also facilitate real-time monitoring and build a more versatile dataset for training deep learning models.

Lastly, exploring advanced deep learning models with video processing capabilities represents a frontier in solar forecasting research. Most current systems rely on still images taken at discrete time intervals, potentially missing the dynamic transitions and temporal context of cloud motion. By treating the input as a video sequence, models such as 3D Convolutional Neural Networks (3D-CNNs), Convolutional LSTMs, or Transformer-based architectures can learn temporal dependencies and movement patterns of clouds across frames. This can lead to significantly better anticipation of changes in solar irradiance caused by transient weather events. Furthermore, such models can be optimized to run in near-real-time, making them highly suitable for deployment in smart grid systems and solar energy management platform.

References

[1] Chodakowska, Ewa, et al. *Solar Radiation Forecasting — State of Art and Methodological Discussion*. 2024.https:https://www.preprints.org/manuscript/202406.0128/v1

[2] Nikitha, M., et al. *Solar PV Forecasting Using Machine Learning Models*. IEEE, 2022. https://ieeexplore.ieee.org/document/9742889

[3] Kosmopoulos, Panagiotis. *Solar power monitoring and forecasting*. ScienceDirect, 2024.https://www.sciencedirect.com/science/article/abs/pii/B9780128233900000041?via%3Dihub

[4] Abdou, Farah Anwar Mohamed, and Muhammad Irfan Memon. *Comparative Analysis on Solar Energy Forecasting Using Random Forest, XGboost, ARIMA, and Different Neural Networks*. IEEE, 2023. https://ieeexplore.ieee.org/document/10584910/authors#authors

[5] Selvi, K. Tamil, et al. *Solar Energy Forecasting for Devices in IoT Smart Grid*. Wiley, 2023. https://onlinelibrary.wiley.com/doi/10.1002/9781119812524.ch13

[6] Nie, Yuhao, et al. *SKIPP'D: A SKY IMAGES AND PHOTOVOLTAIC POWER GENERATION DATASET FOR SHORT-TERM SOLAR FORECASTING*. Cornell University, 2022. https://arxiv.org/pdf/2207.00913

[7] Suraj Chand. *Medium-Term Forecasting of Solar Radiation Using Hybrid Modelling*.Crimson,2023.https://crimsonpublishers.com/prsp/pdf/PRSP.000534.pdf

[8] Rajagukguk, Rial A., et al. *A Deep Learning Model to Forecast Solar Irradiance Using a Sky Camera*. MDPI, 2021.https://www.mdpi.com/2076-3417/11/11/5049

[9] Park, Seongha, et al. *Prediction of Solar Irradiance and Photovoltaic Solar Energy Product Based on Cloud Coverage Estimation Using Machine Learning Methods*. MDPI, 2021.https://www.mdpi.com/2073-4433/12/3/395

[10] Nie, Yuhao, et al. SKYGPT: PROBABILISTIC SHORT-TERM SOLAR FORECASTING USING SYNTHETIC SKY VIDEOS FROM PHYSICS-CONSTRAINED VIDEOGPT. Cornell University, 2023.https://arxiv.org/pdf/2211.14709

[11] Nie, Yuhao, et al. Open-Source Ground-based Sky Image Datasets for Very Short-term Solar Forecasting, Cloud Analysis and Modeling: A Comprehensive Survey. 2022.https://arxiv.org/pdf/2211.14709

[12] Nie, Yuhao, et al. Sky image-based solar forecasting using deep learning with multi-location data: training models locally, globally or via transfer learning. 2022.https://arxiv.org/pdf/2211.02108

[13] Paletta, Q., and J. Lasenby. *Convolutional Neural Networks applied to sky images for short-term solar irradiance forecasting*. Cornell University, 2020. https://arxiv.org/pdf/2005.11246

[14] Al-lahham, Anas, et al. *Sky Imager-Based Forecast of Solar Irradiance Using Machine Learning*. MDPI, 2020.https://www.mdpi.com/2079-9292/9/10/1700

[15] Papatheofanous, Elissaios Alexios, et al. *Deep Learning-Based Image Regression for Short-Term Solar Irradiance Forecasting on the Edge*. MDPI, 2022.https://www.mdpi.com/2079-9292/11/22/3794

[16] Chi Kuo, Wen, et al. *Deep Learning Neural Networks for Short-Term PV Power Forecasting via Sky Image Method*. MDPI, 2022.https://www.mdpi.com/1996-1073/15/13/4779

[17] Hendrikx, N.Y., and K. Barhmi. *All sky imaging-based short-term solar irradiance forecasting with Long Short-Term Memory networks*. ScienceDirect, 2024. https://www.sciencedirect.com/science/article/pii/S0038092X24001579?via%3Dihub

[18] Ghimire, Sujan, and Ravinesh C. Deo. *Deep solar radiation forecasting with convolutional neural network and long short-term memory network algorithms*. ScienceDirect, 2019. https://www.sciencedirect.com/science/article/abs/pii/S0306261919312152?via%3Dihub

[19] Subathra, et al. *Arduino Uno-Powered Parking Guidance System with Ultrasonic Sensors*. IEEE, 2023. https://ieeexplore.ieee.org/document/10435209

[20] Sumathy, R., et al. *Improving Security with Smart Door Lock Access Log using ESP32*. IEEE, 2024. https://ieeexplore.ieee.org/document/10776900

[21] Yakub, Mohd Priyanshu, et al. *Presence Detection with Wi-Fi Using ESP32*. IEEE, 2024. https://ieeexplore.ieee.org/document/10459433

[21] Yakub, Mohd Priyanshu, et al. *Presence Detection with Wi-Fi Using ESP32*. IEEE, 2024. https://ieeexplore.ieee.org/document/10459433

[22] Ruffieux, David, et al. *A 32 kHz DTCXO RTC Module with an Overall Accuracy of ±1 ppm and an All-Digital 0.1 ppm Compensation-Resolution Scheme*. Springer Nature, 2017. https://link.springer.com/chapter/10.1007/978-3-319-61285-0_9

[23] Ahmad, Yasser Asrul, and Teddy Surya Gunawan. *On the Evaluation of DHT22 Temperature Sensor for IoT Application*. IEEE, 2021. https://ieeexplore.ieee.org/document/9467147

[24] Sushanth, T. Sai, and T. Sanjay Siddarda. *Time Series Forecasting using RNN*. IJSREM e- Journal, 2024.https://ijsrem.com/download/time-series-forecasting-using-rnn/

[25] Okut, Hayrettin. *Deep Learning for Subtyping and Prediction of Diseases: Long-Short Term Memory*. IntechOpen, 2021.https://www.intechopen.com/chapters/75265

[26] LeCun, Yann, et al. *Deep learning*. Nature, 2015. https://www.nature.com/articles/nature14539

[27] Al-Malah, Kamal. *Recurrent Neural Network (RNN)*. Wiley, 2023. https://onlinelibrary.wiley.com/doi/10.1002/9781394209118.ch9

[28] Cheng, Zhengzhe, and Hanyi Xu. *Time Series Forecasting Based on ARIMA and LSTM Models*. IEEE, 2023.https://ieeexplore.ieee.org/document/10414963

[29] Cheng, Zhengzhe, and Hanyi Xu. *Time Series Forecasting Based on ARIMA and LSTM Models*. IEEE, 2023.https://asmedigitalcollection.asme.org/OMAE/proceedings-abstract/OMAE2022/85956/V010T11A008/1148164

[30] Kong, Yaxuan, et al. *Unlocking the Power of LSTM for Long-Term Time Series Forecasting*. Cornell University, 2024. https://arxiv.org/pdf/2408.10006

[31] Wang, Yiyang, et al. *Is the LSTM Model Better than RNN for Flood Forecasting Tasks? A Case Study of HuaYuankou Station and LouDe Station in the Lower Yellow River Basin*. MDPI, 2023. https://www.mdpi.com/2073-4441/15/22/3928

[32] O'Shea, Keiron, and Ryan Nash. *An Introduction to Convolutional Neural Networks*. 2015. https://arxiv.org/pdf/1511.08458

[33] Yi, Bongsoo, et al. *Convolutional Neural Networks: An Introduction*. Wiley, 2024. https://onlinelibrary.wiley.com/doi/10.1002/9781118445112.stat08393

[34] Zhao, Haitao, et al. *Neural-Network-Based Feature Learning: Convolutional Neural Network*. Springer Nature, 2020. https://link.springer.com/chapter/10.1007/978-3-030-40794-0_11