Array 27 (2025) 100474

Contents lists available at ScienceDirect

Array

journal homepage: www.elsevier.com/locate/array

IoT-driven real-time weather measurement and forecasting mobile application with machine learning integration

Jul Jalal Al-Mamur Sayor [a,d], Nishat Tasnim Shishir [a,d], Bitta Boibhov Barmon [a], Sumon Ahemed [a], Md. Moshiur Rahman [b,c,*]

[a] Department of IoT and Robotics Engineering, Gazipur Digital University, Kaliakair, Gazipur, 1750, Bangladesh
[b] Department of Software Engineering, Gazipur Digital University, Kaliakair, Gazipur, 1750, Bangladesh
[c] Software Architecture and System Design Lab, Gazipur Digital University, Kaliakair, Gazipur, 1750, Bangladesh
[d] Department of Computer Science and Engineering, Daffodil International University, Daffodil Smart City, Birulia, Savar, Dhaka 1216, Bangladesh

ARTICLE INFO

Dataset link: https://github.com/Software-Machine-Intelligence-Lab/Weather-Station

Keywords:
Weather forecasting
Machine learning
Internet of things (IoT)
Weather data collection
Predictive analytics
Mobile application
Flutter

ABSTRACT

The importance of accurate and timely weather information cannot be overstated, as it is crucial for daily activities, safety, and decision-making across various sectors. Existing weather forecasting systems often lack the precision required for localized conditions, relying on data from distant weather stations and limited environmental parameters. This paper introduces a real-time weather forecasting mobile application that integrates machine learning and IoT technology to address these challenges effectively. The system incorporates a mobile application designed to provide users with real-time weather updates through an intuitive and easy-to-use platform. It utilizes IoT sensors to collect comprehensive environmental data, including temperature, humidity, wind speed, barometric pressure, and rainfall, which are strategically deployed to ensure the collection of localized, high-resolution weather data in real-time. Additionally, the system leverages LoRa technology for robust long-range data transmission. It employs an Incremental Learning model that continuously adapts to new environmental inputs, thereby enhancing forecasting precision and efficiency. APIs (Application Programming Interface) enable efficient data input and retrieval, guaranteeing smooth connection and integration between the sensors and the forecasting algorithms. Moreover, we analyze forecasts from Google and systematically compare them with our localized predictions to highlight the advantages of site-specific deployment for achieving superior localized outcomes. This creative method offers a scalable and flexible solution that can be expanded to cover larger geographic areas in addition to providing precise weather forecasts. The project addresses the limitations of existing weather applications by delivering precise local weather conditions and an intuitive user experience. The initial implementation in Gazipur, Bangladesh, demonstrates the system's effectiveness and potential for wider application nationwide.

1. Introduction

Climate change has increased the unpredictability of weather patterns in recent years, affecting a wide range of industries, including agriculture, construction, transportation, and energy. Accurate and fast weather forecasts are crucial for essential decision-making and operational efficiency in these businesses [1]. For example, agriculture relies on reliable weather predictions for crop planning and irrigation management [2], while transportation and logistics require precise forecasts to enhance safety, minimize delays, and reduce the risk of weather-related disruptions [3,4]. Accurate weather data is crucial for the energy sector, particularly renewable sources like wind, solar, and hydro, as their power generation is heavily dependent on environmental circumstances [5]. Climate change has also increased the frequency and severity of natural disasters like hurricanes, tornadoes, and extreme rainfall events. Real-time weather monitoring is crucial for disaster preparedness and response, minimizing loss of life and property [6]. This stresses the growing importance of accurate, localized meteorological data in enabling educated responses to extreme weather events. Despite advances in meteorological technology, many traditional weather applications still fall short of providing accurate local forecasts. This problem is mostly due to their reliance on faraway weather stations, which frequently fail to detect regional atmospheric variations. Accurately forecasting microclimatic conditions is challenging due to the complex and dynamic character of weather systems [7]. The use of IoT and automation is rapidly increasing across various

* Corresponding author at: Department of Software Engineering, Gazipur Digital University, Kaliakair, Gazipur, 1750, Bangladesh.
E-mail address: moshiur0001@bdu.ac.bd (M.M. Rahman).

https://doi.org/10.1016/j.array.2025.100474
Received 28 September 2024; Received in revised form 18 July 2025; Accepted 18 July 2025
Available online 7 August 2025

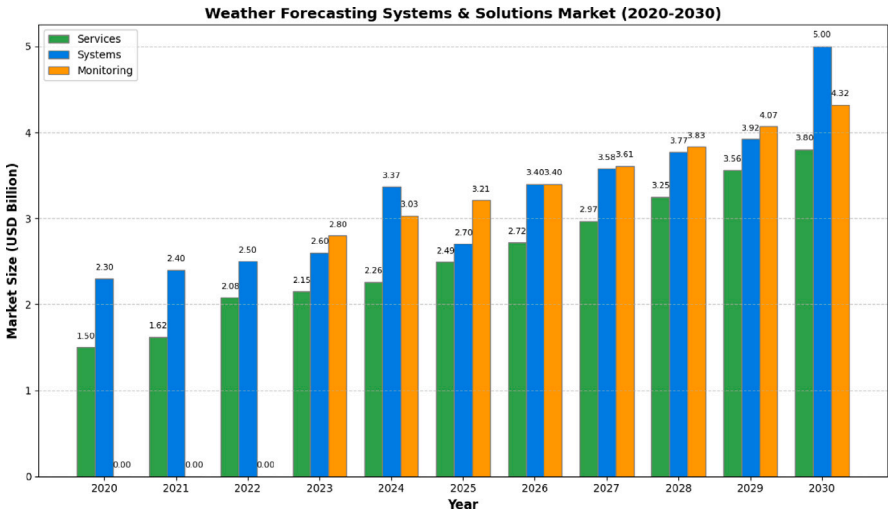

Fig. 1. Projected Market Growth for Weather Services, Systems, and Monitoring (2020–2030).

fields such as agriculture, healthcare [8], restaurants [9], and the hospitality industry, including applications like weather forecasting. But existing weather measuring and forecasting systems often measure only a narrow set of environmental parameters, typically temperature and humidity, while ignoring critical elements such as wind speed, light intensity, barometric pressure, and rainfall [10]. This restricted focus can result in insufficient or misleading weather information, limiting the forecast's dependability and usefulness. These shortcomings highlight the need for a more comprehensive approach that integrates a wide range of environmental data, employs ML for improved prediction accuracy, and offers a user-friendly mobile application to enhance accessibility and reliability in weather forecasting.

The global weather forecasting systems and solutions market is witnessing a steady surge, driven by the growing need for precise climate insights across agriculture, aviation, energy, and disaster management sectors. From a total market size of approximately $3.8 billion in 2020, it has expanded to around $8.4 billion in 2025, with projections reaching nearly $13.1 billion by 2030 [11–13] This robust growth underscores the rising demand for advanced services, sophisticated systems, and real-time monitoring solutions to tackle climate variability and ensure operational efficiency. Fig. 1 illustrates the projected market growth for weather services, systems, and monitoring from 2020 to 2030.

Our work presents a significantly more cost-effective alternative compared to the prices outlined in the [14], where an IoT-based automatic weather station costs thousands of dollars. We developed an IoT-based weather station for approximately 8865 Taka (about 73.88 USD), substantially reducing costs without compromising essential functionality. This paper introduces an innovative weather forecasting system that combines advanced machine learning with IoT technology to overcome the shortcomings of traditional methods. By harnessing real-time data collection, continuous learning, and long-range communication, the system delivers precise, localized weather updates while ensuring scalability and cost-effectiveness. This approach guarantees mitigation of the shortcomings of the existing systems and also keeps the solution deployable efficiently on lower-end hardware, hence adaptive and economically viable under any kind of traffic variation. Furthermore, our contribution aligns with the Sustainable Development Goals (SDGs), particularly SDG 13 (Climate Action) and SDG 11 (Sustainable Cities and Communities), as highlighted in recent Array publications advocating for intelligent climate resilience systems through IoT and ML integration [15].

The objectives of our work can be summarized as follows:

- To collect localized weather data in real time using IoT sensors, ensuring high-resolution and precise environmental monitoring.
- To continuously improve forecasting accuracy by integrating an Incremental Learning model that adapts to new environmental data.
- To achieve robust long-range communication through LoRa technology, enabling reliable data transmission between sensors and the forecasting system.
- To seamlessly integrate system components including IoT sensors, databases, and forecasting models through efficient APIs for smooth, cross-platform operation.
- To develop a scalable, cost-effective solution for precise weather forecasting adaptable to diverse geographical regions.

This work aims to revolutionize weather forecasting by integrating advanced machine learning with IoT technology to deliver precise, real-time localized updates. Ultimately, this scalable, cost-effective approach adapts to diverse geographical regions and varying traffic loads, overcoming the limitations of traditional methods.

The remainder of this paper is structured as follows: Recent related work is discussed in Section 2. Section 3 comprehensively analyzes the proposed method. The implementation details of the system are described in Section 4. Section 5 evaluates the system's performance, including the model's accuracy, simulation procedure, and analysis of the results. Section 6 presents the conclusion of this work. Finally, Section 7 provides the future research directions of our work.

2. Related work

The main objective of our work was to provide a user-friendly mobile application that delivered accurate, real-time and predictive weather forecasts using machine learning (ML), helping users to schedule their plans safely and manage weather-dependent tasks efficiently. In our approach, we developed an IoT system that retrieved real-time data using sensors such as a temperature-humidity sensor, wind speed sensor, barometric pressure sensor, light-dependent resistor, and rain sensor. In this section, we explored several state-of-the-art weather measurement and forecasting techniques.

2.1. IoT-Based weather monitoring systems

Girija et al. [16] propose an IoT-based system for monitoring environmental conditions like temperature, humidity, and CO levels using sensors connected to an ESP8266 microcontroller. The data is sent to the cloud for analysis and can be accessed remotely. The system lacks the integration of advanced data analysis techniques like ML for improved prediction accuracy and anomaly detection. The absence

of a mobile application restricts accessibility and user convenience. Furthermore, the discussion on the accuracy of the sensors and system scalability is lacking, indicating clear areas for improvement. Using standard sensors, Kamble et al. [17] present an IoT-based weather monitoring system that tracks temperature, humidity, wind speed, direction, and rainfall. The data is processed by a microcontroller and updated to a webpage via GPRS, allowing remote access. The system cannot provide real-time continuous updates due to its 10 min web page data refresh interval, which limits the immediacy of weather monitoring. Additionally, the system does not have a mobile application for real-time monitoring and control, reducing its accessibility and usability.

Bulipe et al. [18] propose an IoT-based environmental monitoring system that tracks parameters like temperature, humidity, light, and CO levels using sensors. The data is sent to a web page and displayed as graphical statistics, accessible from anywhere. The system addresses environmental changes by remotely monitoring and controlling conditions, but relies on outdated methods like Zigbee, lacks advanced mobile applications, and may face limitations in accuracy and real-time processing. Piciullo et al. [19] developed an operational IoT-based Local Landslide Early Warning System (Lo-LEWS) for a non-failed steep slope in Eidsvoll, Norway. Their approach integrates real-time hydro-meteorological monitoring with data-driven and numerical modeling to forecast slope stability. A physically based hydrological-geotechnical model using SEEP/W and SLOPE/W (GeoStudio) was calibrated with in-situ sensor observations, including volumetric water content (VWC) and pore water pressure (PWP). These outputs, along with meteorological forecasts and vegetation indices (Leaf Area Index), were used to train Random Forest and Polynomial Regression models for the prediction of the factor of safety (FoS). While the system demonstrates strong modeling, forecasting, warning, and monitoring integration, it is subject to limitations in effectiveness due to model generalizability, demands on accurate sensor and forecast information, and computational demands associated with real-time incorporation of numerical models into the cloud. Ben Bouallègue et al. [20] presented a comparative assessment that determined the performance of ML-based weather prediction models relative to traditional numerical weather prediction (NWP) models in an operational-like environment. The research used the Pangu Weather ML model, which had been trained on ERA5 reanalysis data, and compared its deterministic forecasting capability with the ECMWF Integrated Forecasting System (IFS). The study identified notable limitations in ML-based forecasts, including excessive smoothing, increased bias with longer lead times, and reduced effectiveness in predicting the intensity of tropical cyclones.

2.2. Integration of machine learning and cloud technologies

Gotmare et al. [21] use Arduino UNO and sensors to track temperature, humidity, pressure, wind speed, and air quality, providing real-time notifications. Its limitations include limited data accuracy and a lack of a mobile application for enhanced accessibility. Nallakaruppan et al. [22] introduced a weather forecasting system that integrates IoT and ML methods such as Decision Tree and Time Series Analysis. The system used sensors connected to a Raspberry Pi to gather weather data, which was then analyzed to enhance the accuracy of forecasts. However, the system has limitations, such as the absence of a mobile application, which could impact accessibility and real-time data updates. Alam et al. [23] present a low-cost IoT-based weather station using Node MCU, Arduino Uno, and various sensors to monitor real-time weather data. The system transmits data to the ThingSpeak web server for remote access. Limitations include potential data transmission delays due to network latency, power consumption issues in extreme conditions, and susceptibility to weather-related disruptions impacting sensor accuracy and communication stability.

Verma et al. [24] developed a real-time weather prediction system using IoT devices and sensors to monitor weather conditions. However, the system has limitations, including a basic ML model with lower accuracy, the absence of integration of additional weather parameters such as wind speed, pressure, and rainfall, and a lack of a mobile application for user-friendly accessibility. Kapoor et al. [25] developed a cloud-based weather station using a Raspberry Pi and sensors to monitor temperature, humidity, and pressure. The system uses Raspberry Pi Zero W boards to collect data, which is transmitted to a central Raspberry Pi 3 and then to the cloud for processing and prediction. However, potential issues include data transfer latency, power consumption in extreme conditions, and susceptibility to disruptions from severe weather.

Bonilla et al. [27] proposed a microservices and modular-based weather station software architecture to enable real-time data acquisition, scalability, and flexibility. Their system, based on Docker containers and JSON configuration, can integrate different sensors and allow remote access. It is based on data loggers with web API support and is not validated for large-scale or high-frequency installations.

2.3. Advanced systems with mobile application and ML integration

Several recent works have addressed the potential of real-time, collaborative weather forecasting through machine learning. For instance, Fowdur and Nazir [28] proposed a system that aggregates weather data from multiple locations to enhance predictive accuracy using a suite of ML algorithms such as CNN, KNN, and Multiple Polynomial Regression. Their work emphasized the performance benefits of collaborative forecasting over traditional single-point models and demonstrated a complete system incorporating mobile, desktop, and web interfaces with cloud and local deployment. This approach closely aligns with our system's multi-point data architecture and objective of accurate, real-time forecasting using lightweight, adaptive machine learning. Kodali et al. [29] describe a low-cost weather monitoring system using a Wemos D1 board and the Arduino platform to retrieve weather data from the Cloud and display it on an OLED screen. However, the system lacks mobile app integration and may have limited functionality in areas with poor network coverage or restricted access to cloud services. Krishna, A., et al. [30] use ANN for precise forecasts but face challenges with sensor calibration, resulting in potential inaccuracies, and it lacks real-time user interactivity via mobile apps. Math et al. [26] proposed an IoT-based weather monitoring system for Precision Agriculture (PA) in India. The system uses low-cost sensors connected to an ESP32 microcontroller. Data is collected, processed locally, and stored on the ThingSpeak IoT platform for visualization and analysis to help farmers make timely decisions. However, the system lacks mobile app integration, potentially limiting accessibility, and ThingSpeak may restrict scalability and advanced analytics compared to modern solutions. Perakis T. et al. [31] address scalability but struggle with power consumption and latency issues, limiting real-time weather data accuracy in larger geographic areas. Satyanarayan et al. [32] introduced a smart weather monitoring system utilizing Raspberry Pi 3 and a variety of sensors to measure temperature, humidity, wind speed, rainfall, and light intensity. The collected data is then uploaded to the cloud and is accessible globally through a mobile application, addressing the challenge of remote weather monitoring. However, the system relies on Raspberry Pi 3 for data processing and communication, which may not be adequate for handling more intricate weather monitoring tasks or large-scale deployments requiring higher computational power and reliability. Additionally, the system lacks integration of advanced weather prediction algorithms. Singh, S. et al. [33] integrate IoT and machine learning for real-time weather prediction. However, it has limitations in data accuracy during extreme weather conditions and lacks comprehensive mobile application features for user accessibility.

Chatrabhuj et al. [34] provided a comprehensive overview of AI applications in agriculture and environmental conservation, emphasizing the need for intelligent systems that address data scarcity, climatic challenges, and decision-making in real time. Building on this, our

Table 1
Comparison with some of the state-of-the-art works.

References	Temperature measurement	Humidity measurement	Wind speed measurement	Light intensity measurement	Barometric pressure measurement	Rainfall status measurement	Database integration	ML integration	Mobile application implementation
Rao et al. [18]	Yes	Yes	No	Yes	No	No	Yes	No	No
Kamble et al. [17]	Yes	Yes	Yes	No	No	Yes	No	No	No
Joseph et al. [10]	Yes	Yes	No	No	No	No	No	No	No
Girija et al. [16]	Yes	Yes	No	No	No	No	No	No	No
Verma et al. [24]	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Gotmare et al. [21]	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No
Nallakaruppan et al. [22]	Yes	Yes	No	No	Yes	Yes	Yes	Yes	No
Kapoor et al. [25]	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	No
Alam et al. [23]	Yes	Yes	Yes	No	No	No	Yes	No	Yes
Math et al. [26]	Yes	Yes	No	Yes	No	Yes	Yes	Yes	No
Proposed work	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

work expands these principles into weather intelligence by merging IoT sensing with incremental machine learning for localized forecasting, a necessity under rapid climate variability. Table 1 compares the related works with our proposed weather measurement and forecasting method. The table highlights key aspects such as the data collection parameters and forecasting techniques.

3. Methodology and system design

Our proposed system employs LoRa (Long Range) wireless communication technology on the transmitter and receiver sides to facilitate secure, low-power, and long-range wireless data transfer. Unlike regular communication systems based on Wi-Fi or mobile networks, LoRa does not depend on internet infrastructure. This makes it highly appropriate for use in rural, remote, and infrastructure-scarce regions where consistent internet connectivity may be unavailable. By offering seamless communications between remote sensor nodes and the central processor without relying on external network services, the system supports seamless data gathering and transmission. The proposed methodology not only enhances system reliability and autonomy but also reduces the operational expense and network deployment complexity in remote sites to a significant extent.

The core of this work is an incremental learning methodology-enabled ML model for forecasting weather data. A flutter-based mobile application is built to showcase the real-time weather data from an IoT system. The overall process of the proposed system is illustrated in Fig. 2. The work is divided into multiple steps:

i. Data collection
ii. Data pre-processing
iii. Model training
iv. Backend
v. Visualization

Elaboration of the steps mentioned above is as follows:

3.1. Data collection

We have developed a system capable of measuring real-time weather parameters such as temperature, humidity, air pressure, light intensity, wind speed, and rain conditions. In addition to monitoring weather conditions, we also forecast these weather parameters to leverage the insights for better decision-making. To enhance the accuracy of our predictions, we supplement real-time data with additional information from reliable websites, ensuring that our forecasts are more precise and dependable. We collect data from www.visualcrossing.com from 2000 to April 2024 [35]. We have collected one record per day from the website.

3.1.1. Ethical considerations and data privacy

This study utilizes environmental weather data collected through IoT sensors and from publicly accessible sources such as Visual Crossing. The data comprises non-personally identifiable parameters including temperature, humidity, wind speed, air pressure, and rainfall. All data collection and handling procedures comply with standard IoT protocols and environmental data management practices. No user-specific or private data are collected, stored, or processed in this research. The collected data are anonymized, securely transmitted, and stored by applicable data privacy standards. No ethical approval was required as part of this study, given that no human subjects or private data were involved.

3.2. Data pre-processing

Before using the data from the source to train our model, we apply several pre-processing methods like dropping unused columns and null data, filling null data, formatting patterns, and Label encoding to make our dataset well-formatted. These pre-processing methods are elaborated as follows:

3.2.1. Dropping unused column

The dataset that we collected from the website had about 33 columns. We selected seven data types: date, maximum temperature, minimum temperature, average temperature, air pressure, weather conditions, wind speed, and humidity. Since we have an IoT system that uses a limited number of sensors, a certain amount of data has been collected from the system. According to the data type, we select columns from the dataset we collect from the website.

3.2.2. Dropping some null data

Our dataset contains 8885 records, with 35 records having null values. Due to the large size of the dataset, we remove these null values to ensure data integrity; the exclusion of these null values does not impact the overall performance.

3.2.3. Filling null data

After removing the null values from the dataset, we discovered that the pressure column still contains many null records. We addressed this issue by calculating the median value from all the pressure records and filling in the null values.

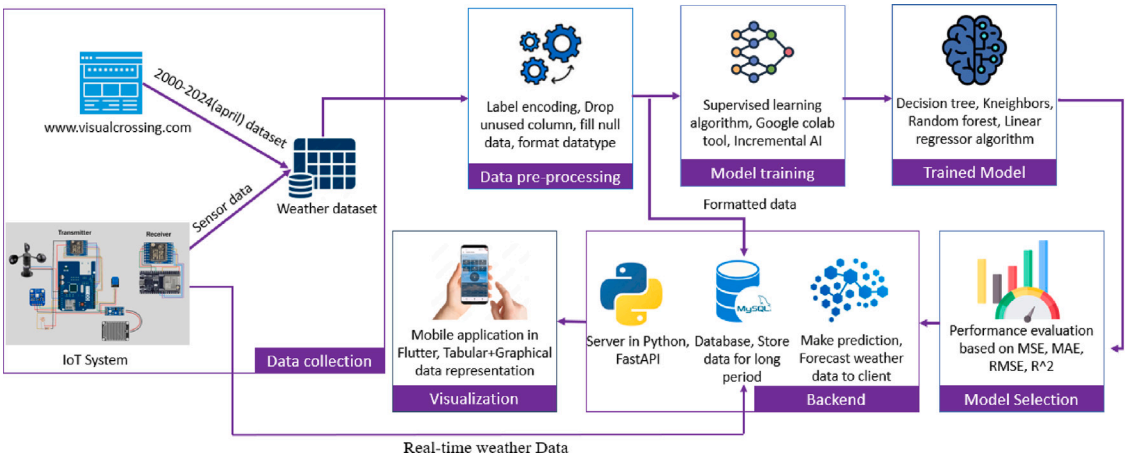

Fig. 2. Overall process of our proposed system.

3.2.4. Format data pattern

After collecting data from the website and our IoT system, we noticed inconsistencies in the date format. The data includes three different patterns: "2000-01-05", "1/1/2016", and "2024-04-22 19:00:27". To standardize these formats, we extracted the day, month, and year values from each date string using the Python methods strptime and to DateTime.

3.2.5. Label encoding

Label encoding is an ML and data analysis technique that converts categorical variables into a numerical format [36]. We had a column in the dataset that described the weather condition in string datatypes, like "Is the day sunny or cloudy?" Since we use a regression ML algorithm that works with numerical values only, we use the label encoding technique to represent these string values as numeric values. For example, "0" values represent sunny, and "1" represents cloudy, etc.

3.3. Model training

For our weather forecasting system, we constructed a custom dataset comprising historical weather observations, which was meticulously cleaned and preprocessed to ensure quality inputs for training. All model development and experimentation were carried out in Google Colab, taking advantage of its scalable and reliable computational environment to support efficient model iteration. The target outputs consist of temperature, humidity, wind speed, and atmospheric pressure. As these are continuous numerical variables, regression algorithms were identified as the most appropriate choice. To select the optimal predictive model, we conducted a comparative analysis of several supervised regression techniques, including Linear Regression, Decision Tree Regressor, K-Nearest Neighbors Regressor, and Random Forest Regressor.

Each model was trained on 80% of the data (train split) and validated on the remaining 20% (test split) to assess generalization capability. During training, we carefully selected and tuned key hyperparameters — such as the number of estimators in the Random Forest (set to 100), maximum tree depth, and minimum samples per leaf — to balance bias and variance and mitigate overfitting. Evaluation was performed using standard metrics: Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and the R-squared (R^2) score, providing a comprehensive assessment of predictive accuracy. Among the tested algorithms, the Random Forest Regressor demonstrated the best performance, achieving the lowest error rates and highest R^2 values across all weather parameters. Its ensemble learning strategy, which aggregates predictions from multiple decorrelated decision trees, proved highly effective in modeling complex, non-linear relationships inherent in weather data [37].

We want to ensure consistent data availability and an improved customer experience, particularly via our mobile application. Given mobile devices' processing capacity and energy efficiency limits, we deliberately chose not to implement resource-intensive models such as Gradient Boosting Machines, XGBoost, or deep neural networks at this moment. Instead, we chose lightweight ensemble models that are easier to deploy on mobile platforms and provide a fair trade-off between accuracy, speed, and efficiency. As our dataset expands and infrastructure advances, future versions will investigate more complicated models using an incremental learning method to improve predictive performance while maintaining mobile usability.

3.3.1. Incremental learning

We have used an advanced ML model training methodology called Incremental learning, where an ML model learns and enhances its knowledge progressively without forgetting previously acquired information [38]. Our developed weather prediction system utilizes an advanced incremental learning model that continuously re-trains itself using both historical and real-time data. This approach ensures that forecasting accuracy improves over time by incorporating data from the initial deployment up to recent periods, such as April 2024. By retraining on an extended dataset, which includes past weather records and newly collected environmental data, the system adapts to changing conditions, maintains high accuracy, and ensures scalable long-term performance. Fig. 3 shows the conceptual flow of our proposed Incremental Learning model.

To improve the accuracy of our machine learning model, we implemented a retraining technique in which the model is updated with a larger dataset. This current dataset was created by adding newly available data to previously utilized training data. As a result, the model's learning capacity rises, resulting in higher prediction accuracy. Our mobile application also includes a function that allows for on-demand model retraining, which promotes incremental learning and keeps the model up to current with changing data trends.

3.4. Backend

For our backend, we used FastAPI, a modern web framework known for its high performance and ease of use [39]. Using this, we make total nine API that provide several services like creating users, reading real-time weather data, and more, as provided in Table 2. We use MySQL as our database, an open-source relational database management system, to store real-time sensor data from IoT hardware systems, historical weather data, and user information. Our mobile application

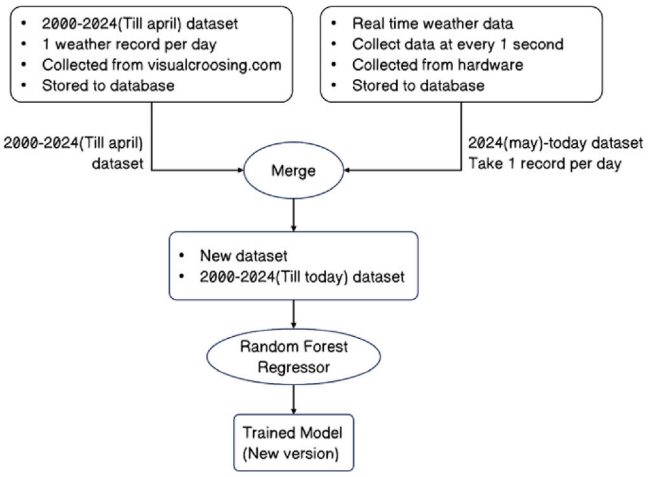

Fig. 3. Conceptual flow of our proposed incremental learning model.

Table 2
List of APIs and their usage in our proposed system.

API name	Purpose
Create a user	Register a new user in the application
Generate OTP	Generate an OTP during user registration or password update
Login	Authenticates the user during login
Forgot password	Update the password using the provided email and OTP
Get user information	Retrieve information of the logged-in user
Insert weather data	Inserts hardware sensor data (temperature, humidity, etc.) into the database
Get the latest weather information	Retrieve the latest weather data for a specific location
Get weather line chart data	Fetches the latest ten wind-speed sensor data for chart plotting
Get weather prediction data	Provide predicted weather data for a specified date

and server interact using the request–response communication model. This pattern follows a synchronous communication approach, where the client (mobile application) sends a request to the server and waits for a response. Through this server, users can access real-time weather data, receive future weather predictions, create an account, log in, reset their password, and access other related services.

3.5. Visualization

In our research, we developed a mobile application using Flutter to provide users with real-time weather data like temperature, humidity, light intensity, air pressure, wind speed, rain status, which are sourced from our IoT system. The application offers both real-time and predicted weather information, presented in a tabular format, with an option for graphical visualization. The notification alert option of the system gives alert to the users when the weather changes suddenly so that the users can get the real time emergency update quickly and take necessary decisions according to the current environmental situation.

All the major visuals from the front-end are added in the implementation section.

4. Implementation

The system is structured into two main units: the receiver (server) and transmitter (sender). The overall architecture of the proposed system and the IoT circuit diagram are illustrated in Figs. 4 and 5, respectively. On the transmitter side, an Arduino Uno microcontroller is utilized, which contains several analog input pins that can easily be connected to various environmental sensors. The system consists of a DHT21 temperature and humidity sensor, a BMP180 barometric pressure sensor for reading atmospheric pressure, an anemometer to monitor wind speed, a rain sensor module for reading rainfall intensity, and a photoresistor (LDR) with a resistor for reading ambient light intensity. All these sensors constantly gather live weather data, which is then transmitted wirelessly using an RA-02 LoRa module interfaced to the Arduino.

On the other end, the system also includes another RA-02 LoRa module connected to an ESP32 microcontroller. The LoRa module collects the data sent and forwards it to the ESP32, which processes and forwards the data to a local server (localhost). With the ESP32's built-in Wi-Fi capabilities, real-time data visualization and storage are facilitated without relying on external cloud services or an internet connection. This design will maintain the system completely operational in remote or rural regions with limited infrastructure and offer high scalability and accuracy in collecting and processing local weather information.

4.1. Material used

In our project, we have used a range of sensors and modules to enable proper and accurate environmental monitoring. Arduino Uno is selected due to its simplicity, versatility, and compatibility with several sensors as the main microcontroller of the sensor node. To monitor weather parameters, we have employed the use of the DHT21 sensor for precise measurement of temperature and humidity and the BMP180 for high-accuracy barometric pressure measurement. The FC-37 rain sensor is utilized to detect rain with analog and digital outputs for convenient integration. An anemometer is utilized to accurately detect wind speed, which is crucial in acquiring full weather data. The LDR is utilized to detect light intensity, which varies resistance according to the ambient light, thus suitable for monitoring light conditions. Each component is chosen for dependability, accuracy, and low power consumption, which supports the system's purpose of long-term, real-time monitoring within the external world.

Within the system architecture, we used an Arduino Uno with an RA-02 LoRa module on the transmission side and an ESP32 with another RA-02 LoRa module on the receiver side. Arduino is chosen as the transmitter because of its ease of sensor interfacing and low power consumption with consistent performance, making it most suitable for remote data acquisition. LoRa is employed because of its low-power wireless communication capabilities for distances and supporting consistent data transmission irrespective of cellular or Wi-Fi coverage, which is most critical for rural areas. On the receiver side, ESP32 is chosen because it offers more processing power and integrated Wi-Fi capabilities, enabling efficient receipt of data from the LoRa module and seamless communication to a cloud server or local display interface. This separation enables a cost-effective, reliable, and scalable approach for remote environmental monitoring.

4.2. Hardware implementation

In Fig. 6, the LoRa and ESP modules were deployed, the sensors were connected to an Arduino, enabling the collection of real-time environmental data, which was then transmitted and stored for subsequent analysis.

4.3. Hardware cost

This section describes the necessary tools and the total cost of one station. In Table 3, we calculate the total cost in BDT for hardware and maintenance according to a shop named "Robotics Bangladesh"[40] and added the equivalent cost in USD. The overall cost is about 8865 Taka in BDT and 73.88 dollars in USD

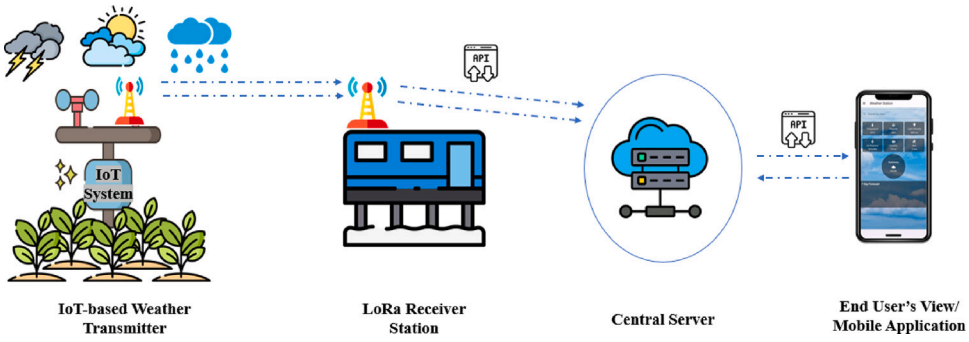

Fig. 4. Operational workflow of the proposed IoT-based weather monitoring system.

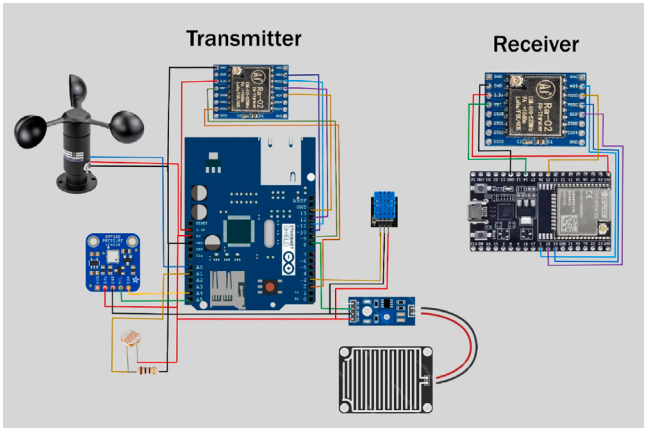

Fig. 5. Circuit diagram of the proposed IoT-based weather monitoring system.

Table 3
Hardware cost of IoT system development.

Tools	Quantity	Cost (BDT)	Cost (USD)
Arduino Uno	1	1050	9.76
Uploader Cable for Arduino	1	150	1.39
LoRa SX1278 (Sender)	1	875	8.13
LoRa SX1278 (Receiver)	1	875	8.13
ESP32	1	650	6.04
DHT21	1	550	5.11
Anemometer	1	2950	27.42
BMP-180	1	290	2.70
Rain Sensor	1	145	1.35
LDR	1	30	0.28
Resistor (10 kΩ)	1	10	0.09
Router	–	1000	9.30
Breadboard	2	90	0.84
Jumping Wires	60	200	1.86
Total cost	–	**8865**	**73.88**

4.4. Mobile application

In this work, we developed a mobile application that provides real-time weather information as well as future weather predictions. The application was built using the Flutter framework for front-end development, while the back-end was implemented using FastAPI to ensure efficient data handling and processing. The workflow of our developed mobile application is shown in Fig. 7.

The very first screen of our Flutter-based mobile application is a splash screen showcasing a logo. The splash screen takes about 1 s; within this time, the mobile application retrieves the GPS permission for location extraction. After the splash screen, the users are directed to the registration page. Suppose the user is new to the system. In that case, they must register by providing information such as name, email, password, and role or if the user already has an account, they need to press the login button, which will direct them to the login page. Additionally, we have used form validation in the registration input fields and an OTP-based email verification for registration. Without fulfilling proper validations, users cannot register successfully. After submitting the registration page, an OTP will be sent to the given email of the users; users have to put the exact OTP in the dialog box of the mobile application. If the OTP provided by the users is matched, they can proceed to further app activities.

The OTP verification flow is shown in Fig. 8. From the Login page, Only the registered users can log in to the mobile application by giving their registered username and password as shown in Fig. 9. Another essential feature of our mobile application is password resetting. Users can reset their password if they forget it or for any other security issues by simply giving their email and the new password, as shown in Fig. 10. After providing the required information, an OTP will be sent to the user's email. Then, users have to put the correct OTP in the dialog box. If the OTP matches the sent one, the new password will be reset.

After completing the registration and login process, the user will be directed to our mobile application's Home page, which looks like Fig. 11. Users can find several features on the homepage, such as weather station search, real-time weather data and summary, graphic data visualization shown in Fig. 12, and ML-based weather forecasts. Moreover, from the bottom navigation bar, users get important features like location retrieval by clicking on the location icon. Users can get visual graphs of weather data by clicking on icons like wind speed, temperature, etc. From the IoT hardware system, we showcase the real-time, precise weather data on our home screen, shown in Fig. 11. Six boxes on the home page provide real-time weather readings like temperature, humidity, wind speed, light intensity, air pressure, and rain conditions. In the summary section, users can find the weather overview, whether sunny or rainy. We have set a condition that if the temperature is above 30, it will show sunny; if it is raining, the weather summary will be rainy. From the weather forecast table, our mobile app will help users to know futuristic weather information such as average temperature, pressure, and humidity. Users can set a date, and the app will give a three-day forecast about the weather.

The other functionality included in the system is the real-time notification alert, as shown in Figs. 13, 14, within the mobile application. As soon as any parameter exceeds a set threshold, the application sends the user an automatic notification alert. This functionality raises users' awareness and enables them to take prompt measures against excessive temperatures, which contributes to improved safety and decision-making. Users can get more options like dark mode, edit profile, settings, and logout by clicking on the drawer icon at the top left corner of our home page. The complete code for our project, including the frontend, backend, and database, is available at https://github.com/Software-Machine-Intelligence-Lab/Weather-Station.

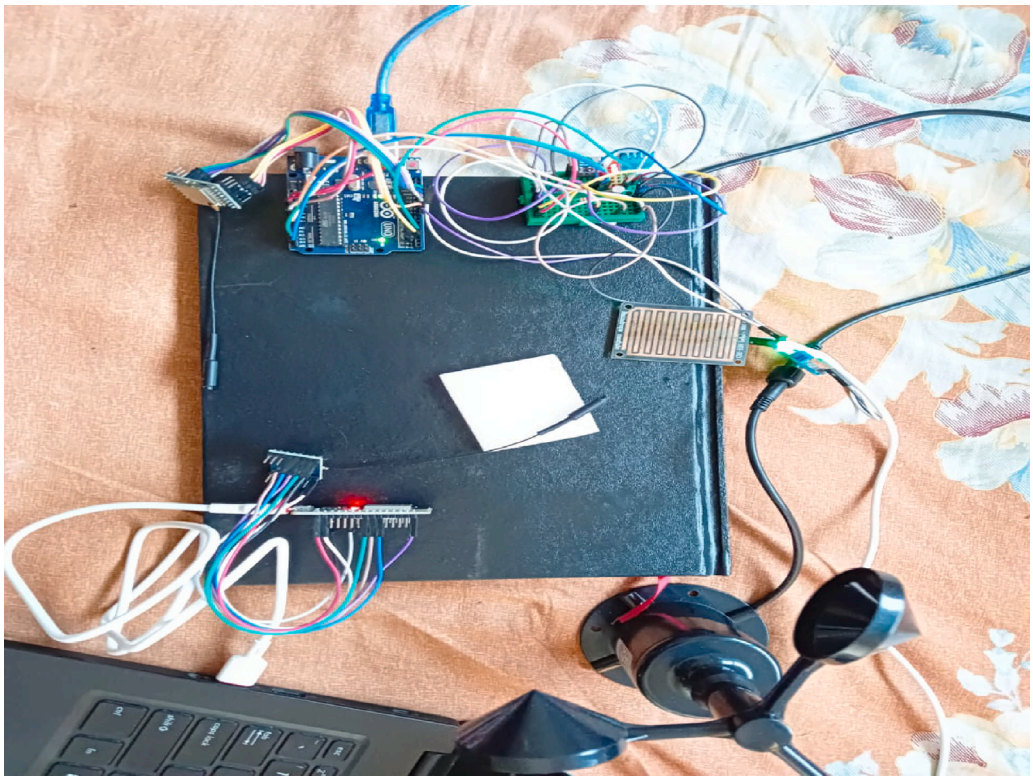

Fig. 6. Functional prototype of proposed system.

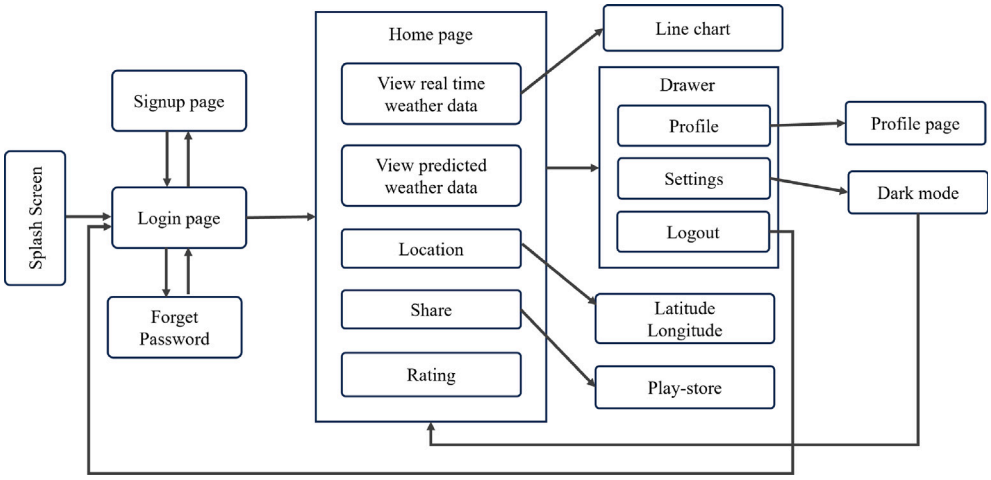

Fig. 7. Workflow of the developed mobile application.

5. Result and discussion

This chapter presents a comprehensive evaluation of the developed weather monitoring and forecasting system, detailing both the implementation environment and the performance outcomes. The mobile application was created using Flutter and Android Studio, supported by a real-time hardware setup involving Arduino and various sensors. To ensure accurate weather forecasting, multiple machine learning regression models were tested and compared using standard performance metrics. The results, along with visual comparisons of predicted versus actual values, are discussed in depth to highlight the effectiveness and reliability of the system.

5.1. Simulation environment

In order to create the mobile application for this research project, we used Android Studio (version 2024.1.2) together with the Flutter Software Development Kit (SDK) 3.22 (current version: 3.24). The Flutter SDK v3.22 allows the application to be used on smartphones running Android 5.0 (Lollipop) or higher. The Arduino Uno hardware component, with 32 KB of storage, was used to collect data from sensors and transfer it to the given API for further processing. We created a hardware setup with the essential sensors connected to an Arduino to collect environmental data in real time and save it in a database using an Ethernet shield. We used the appropriate APIs to connect the database and the mobile application. We want to install weather stations around Bangladesh to improve the mobile app's accuracy and

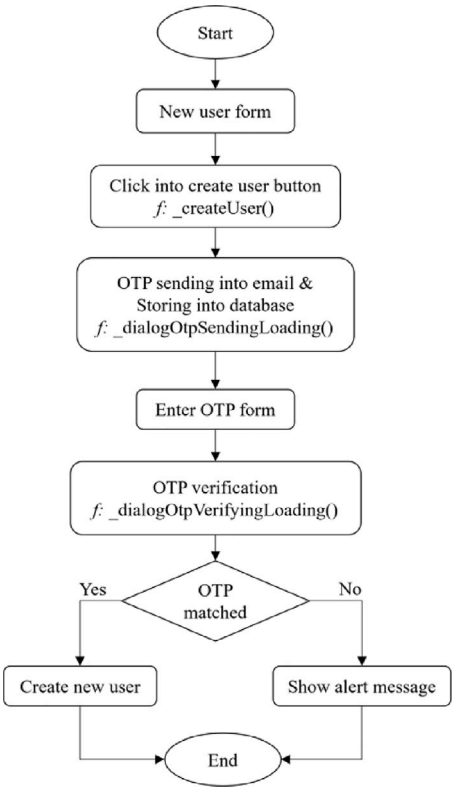

Fig. 8. OTP verification flow of our proposed mobile application.

localization. Users just register and log in with their registration information in the app, and then search for a weather station in their area to get accurate and dependable localized weather information. Users can also view weather forecasts based on past data.

5.2. Performance index

To evaluate and choose the best supervised learning algorithm for our bespoke weather prediction model, we used three essential metrics: Mean Squared Error (MSE), Mean Absolute Error (MAE), and Root Mean Squared Error. These metrics provide a full evaluation of the model's performance, allowing for more precise comparisons of algorithm effectiveness.

5.2.1. Root Mean Square Error (RMSE)

The RMSE calculates the average magnitude of errors between expected and actual values. Lower RMSE indicates a better fit of the model to the data [41]. Because the errors are squared before averaging, greater errors have a disproportionately high influence, making RMSE susceptible to outliers.

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

5.2.2. Mean Square Error (MSE)

MSE measures the average of the squared errors between predicted and actual values. Like RMSE, a lower MSE indicates a better model. MSE is useful for understanding the variance of the prediction errors [42]. Squaring the errors makes MSE sensitive to large errors, but unlike RMSE, it does not return to the original scale of the data.

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

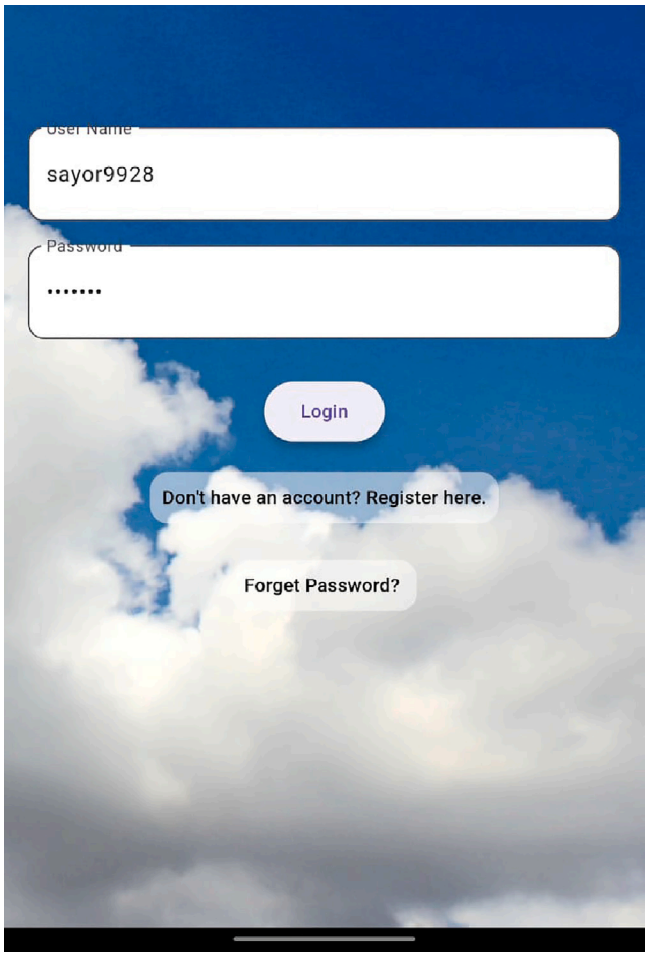

Fig. 9. Login page.

5.2.3. Mean Absolute Error (MAE)

MAE measures the average magnitude of the errors in a straightforward way. MAE is less sensitive to outliers than RMSE and MSE because it does not square the errors. It provides a linear score, which means that all individual differences are weighted equally on average [43].

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{3}$$

5.2.4. Coefficient of determination (R^2)

R^2, also known as the coefficient of determination, measures the proportion of variance in the dependent variable that is predictable from the independent variables. It indicates the goodness-of-fit of a model and ranges from 0 to 1, where a value closer to 1 indicates a better fit. Unlike error-based metrics, R^2 evaluates how well future samples are likely to be predicted by the model.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{4}$$

Here, y_i represents the actual values, $\hat{y}_i$ the predicted values, and $\bar{y}$ the mean of the actual values.

5.3. Model performance

We aim to predict continuous values, such as temperature, humidity levels, wind speed, or any other numerical weather parameter. Regression is appropriate when the output is a constant quantity rather than a category or class. Table 4 presents a comparison of four regression

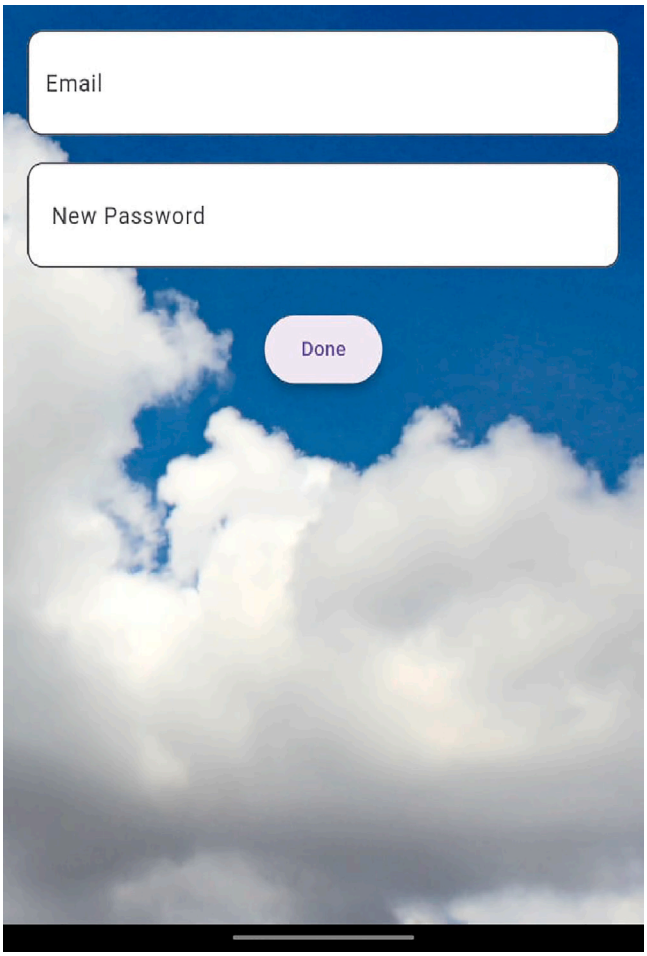

Fig. 10. Forget password page.

Table 4
Performance metrics for different regression models.

Model	MSE	MAE	RMSE	R^2 (Accuracy)
DecisionTreeRegressor	33.63	2.30	5.80	0.8655 (86.6%)
KNeighborsRegressor	45.85	3.77	6.77	0.8166 (81.7%)
RandomForestRegressor	**23.09**	**1.97**	**4.81**	**0.9076 (90.8%)**
LinearRegression	197.26	8.78	14.04	0.2109 (21.1%)

models based on our evaluation metrics. The Random Forest Regressor emerged as the top performer, exhibiting the lowest errors with an RMSE of 23.0911, an MSE of 1.9711, and an MAE of 4.8053. Consequently, we selected the Random Forest Regressor for our system to predict weather information.

The graphs shown in Figs. 15, 16, 17, 18 depict a comparison between actual and predicted values of several environmental parameters like temperature, humidity, wind speed, and pressure over a selected period that is January 2020. The blue line represents the actual recorded values, while the red dashed line shows the predicted values generated by our custom-trained model. Observing the lines, the model 's predictions closely follow the trend of the actual values, demonstrating its ability to capture the overall pattern and fluctuations. Although there are some discrepancies, the predicted values remain relatively close to the actual measurements. These graphs show that the model performs well in predicting several environmental parameter values. However, fine-tuning the model could improve accuracy, reducing the differences observed in specific instances.

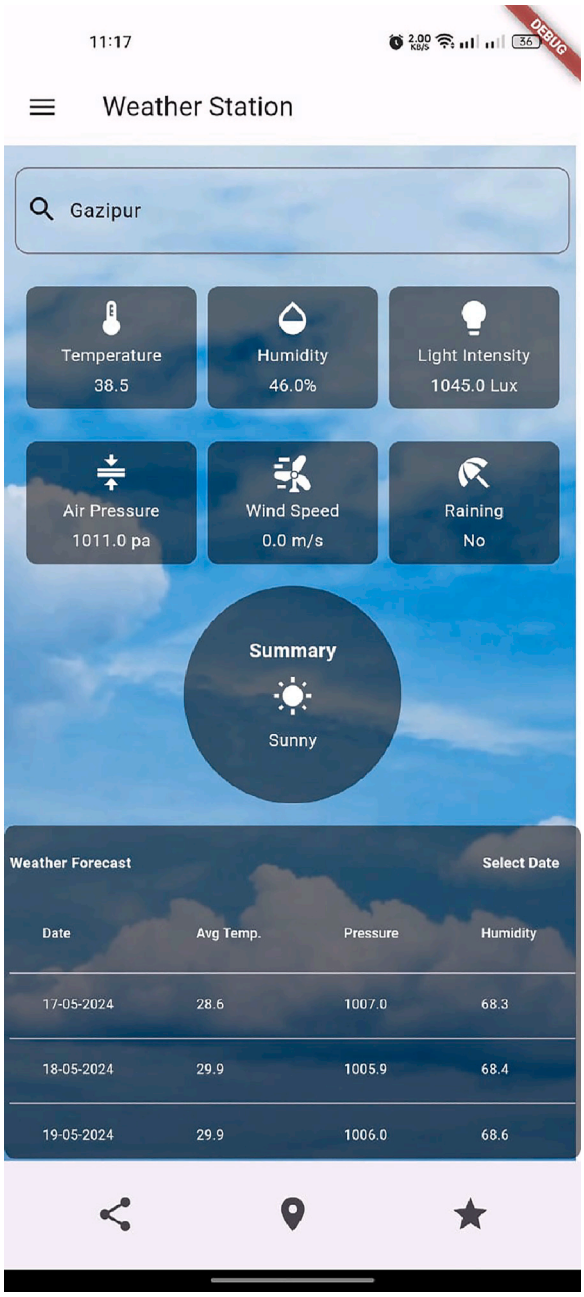

Fig. 11. Homepage.

5.4. Results analysis individual parameters

In this project, we developed a Flutter-based mobile application designed to monitor and forecast weather data, aiming to minimize weather-dependent challenges. The application has been successfully tested multiple times, demonstrating its ability to efficiently fetch real-time data from a cloud database, sourced from our custom hardware setup. This setup, which accurately measures various environmental parameters, is cost-effective with a total hardware expense of 8865 BDT, making it practical and suitable for real-world applications.

The application integrates a machine learning model to forecast weather parameters. Four regression models, Random Forest Regression, Decision Tree Regression, K-nearest neighbors Regression, and linear regression, were evaluated. Random Forest Regression outperformed the others, achieving the lowest Mean Squared Error (MSE:

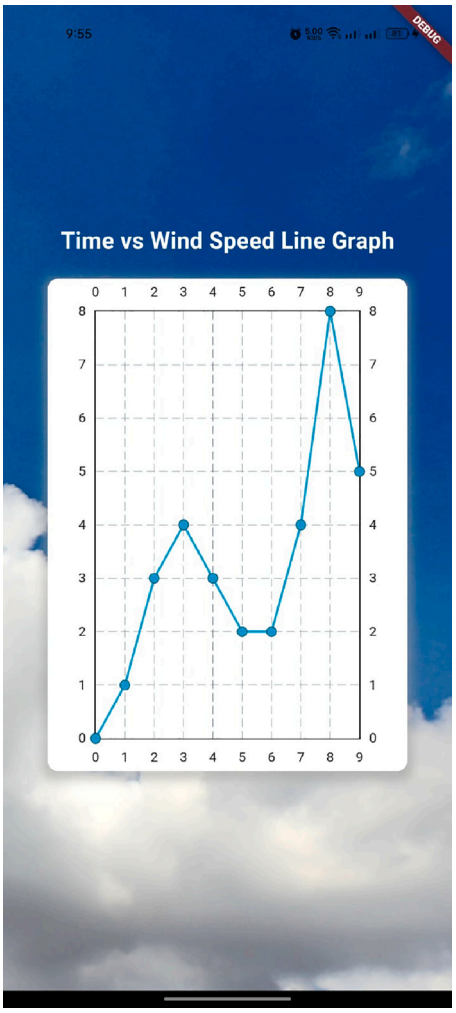

Fig. 12. Real-time weather data visualization in the mobile application.

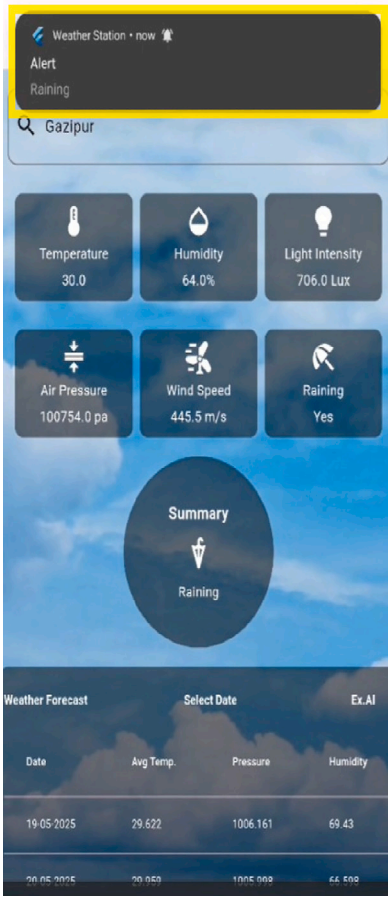

Fig. 13. Notification alert.

Table 5
Performance evaluation of weather parameter predictions.

Parameter	MAE	RMSE	R^2 score
Temperature (°C)	0.56	0.74	0.91
Humidity (%)	1.24	1.60	0.88
Wind speed (m/s)	0.72	0.95	0.87
Pressure (HPa)	0.81	1.02	0.89

23.09), Mean Absolute Error (MAE: 1.97), and Root Mean Square Error (RMSE: 4.80), significantly outperforming the other models tested. To evaluate the prediction accuracy for individual weather parameters, we assessed the final model using MAE, RMSE, and R^2 scores. As presented in Table 5, temperature prediction achieved the highest accuracy with an R^2 of 0.91 and the lowest MAE of 0.56 °C. Humidity, wind speed, and pressure also exhibited strong performance with R^2 scores of 0.88, 0.87, and 0.89, respectively. The corresponding RMSE values were 0.74 °C (temperature), 1.60% (humidity), 0.95 m/s (wind speed), and 1.02 HPa (pressure), indicating reliable and consistent predictions across all parameters. These results validate the robustness and effectiveness of the integrated forecasting system.

5.5. Localized forecast performance evaluation

Fig. 19 presents a 24-h comparison for June 10, 2025, capturing hourly variations in temperature. This figure demonstrates the capability of our system to closely track real-time local conditions throughout

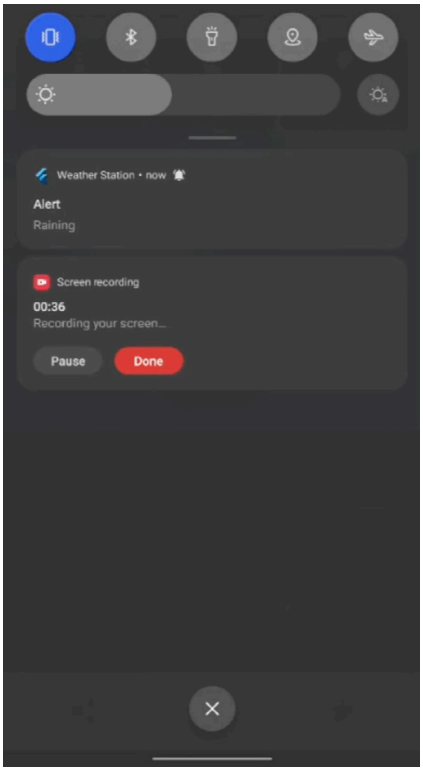

Fig. 14. Alert in notification bar.

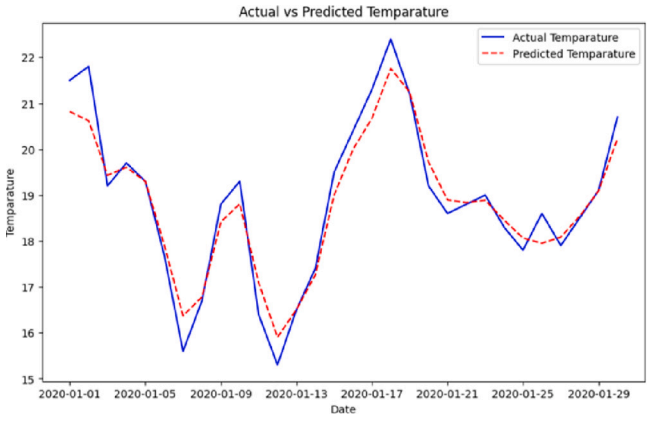

Fig. 15. Actual vs. Predicted temperature.

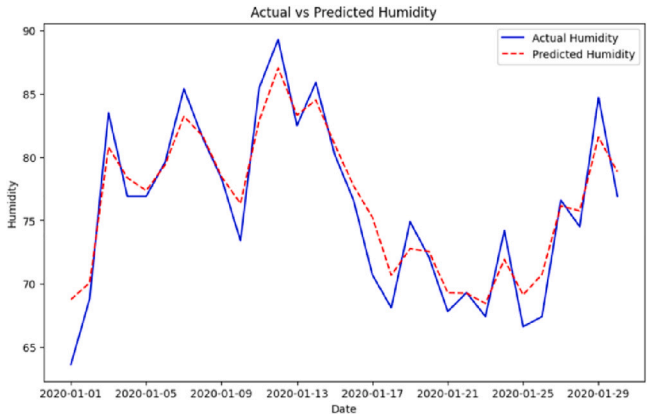

Fig. 16. Actual vs. Predicted humidity.

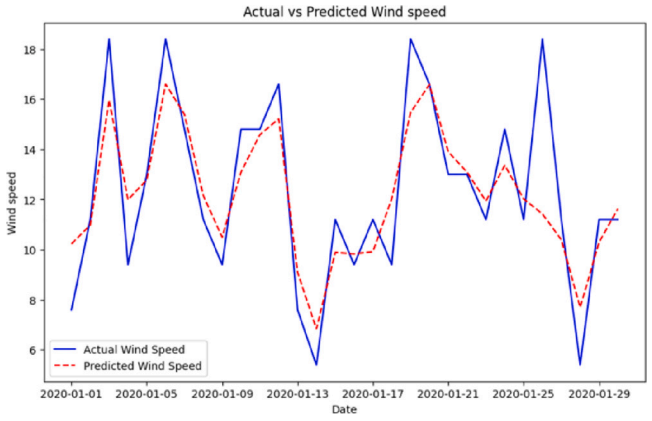

Fig. 17. Actual vs. Predicted wind speed.

the day and emphasizes the advantage of integrating localized IoT data with machine learning for forecasting. These analyses underscore the value of deploying localized sensor networks coupled with tailored machine learning models for significantly improved forecasting accuracy, especially in regions where micro-climatic variations can differ substantially from broader regional forecasts.

The Fig. 20 displays a 7-day temperature forecast comparison from June 11–17, 2025, for BDU Boys Hall, Kaliakoir, Gazipur. It shows three data series: "Our Model Forecast" (dashed blue line), "Sensor Data" (green line), and "Google Forecast" (orange line). While the Google Forecast provides a generalized forecast for the Kaliakoir, Gazipur region with no major deviations, the "Sensor Data" represents measurements from a localized sensor deployed specifically at BDU Boys Hall, Kaliakoir, offering a more precise local temperature reading. The "Our Model Forecast" appears to track the sensor data closely, suggesting an attempt to provide a localized prediction.

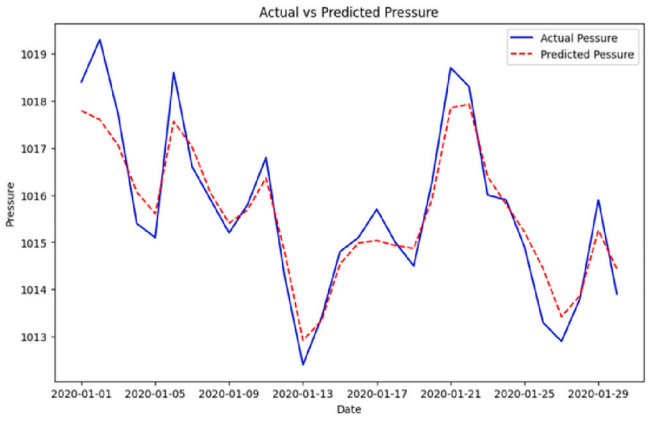

Fig. 18. Actual vs. Predicted pressure.

5.6. Impact of incremental model on model performance

Since weather datasets are often behind paywalls, we sourced our historical weather data from the freely accessible Visual Crossing platform. To maintain consistency and feasibility, we collected one data entry per day spanning from the year 2000 to 2024, resulting in a gradually expanding dataset. To evaluate how the volume of data influences model performance, we trained and tested our machine learning models on incrementally larger subsets of this dataset. The results indicate that as the amount of training data increases, the prediction accuracy of the model improves. Specifically, metrics such as MSE, MAE, and RMSE consistently decreased with the inclusion of more historical data, demonstrating the model's enhanced ability to recognize underlying trends and seasonal patterns.

Fig. 21 shows that with the increment of training data, the performance of the model significantly improves. For these reasons, we developed our weather prediction model with an incremental learning approach. This design allows the model to adapt and improve continuously as more data becomes available over time, ensuring that its predictions become increasingly accurate and reliable. Such a strategy not only optimizes performance but also supports scalability and long-term applicability for real-world deployment.

Several prior studies have explored weather forecasting using IoT-integrated systems. For instance, in the work titled "Weather Forecasting Method from Sensor Transmitted Data for Smart Cities Using IoT", Sreenivasulu Bolla et al. [44] reported an accuracy of approximately 85% based on a dataset containing 7000 instances. Similarly, Abhishek et al. [45] in their study "Weather Forecasting using Machine Learning Algorithms" achieved an accuracy of around 87% using decision tree and linear regression models on sensor-generated data. In contrast, our proposed system achieves an improved accuracy of over 90%, leveraging an incremental learning framework combined with real-time IoT data streams and a continuously expanding dataset. This enhancement demonstrates the increased adaptability, scalability, and precision of our system in dynamic and evolving environmental conditions.

5.7. Evaluation of user-friendliness and decision-making capabilities

The developed system is designed with a focus on user-friendliness and practical decision-making support. The Flutter-based mobile application offers real-time, location-specific weather updates through

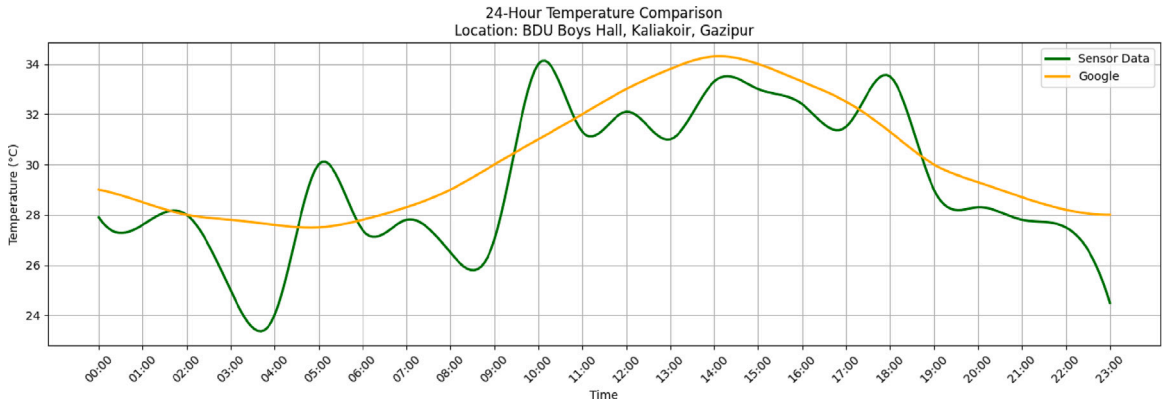

Fig. 19. Real-time hourly weather tracking on June 10, 2025: Localized Sensor vs. Google.

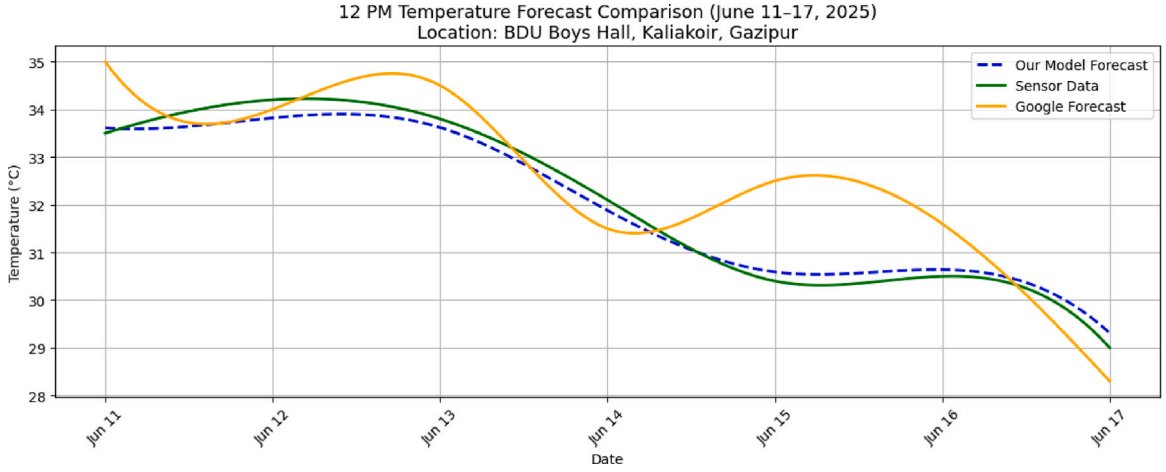

Fig. 20. 7-Day forecast validation using local sensor and external forecast data. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

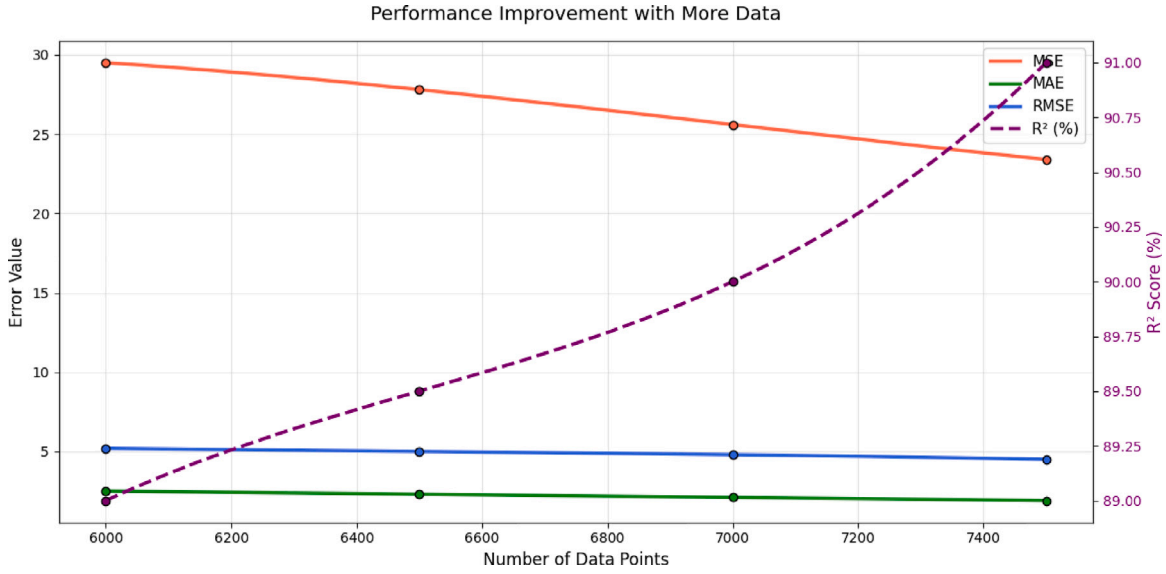

Fig. 21. Impact of incremental data on model performance.

an intuitive and accessible interface, supporting features such as dark mode to enhance usability and comfort across diverse user groups. Timely notifications alert users to sudden weather changes, enabling them to make quick, informed decisions — such as planning outdoor activities or implementing safety measures — based on accurate and localized forecasts.

Graphical and tabular representations of weather trends further improve understandability, allowing users to quickly interpret complex data and make decisions with confidence. The system benefits from exceptionally low latency, as it is entirely self-developed, minimizing external processing delays. Moreover, by employing LoRa communication for sensor data transmission, it ensures ultra-low latency and robust performance even over long distances without depending on conventional internet infrastructures. This makes real-time data retrieval highly responsive, critical for immediate, location-sensitive decision-making.

Additionally, the application serves essential sectors like agriculture and disaster management by providing scalable, precise, and context-specific weather intelligence, thereby enhancing operational safety and efficiency. The continuously learning machine learning models maintain and improve prediction reliability over time. Altogether, the system seamlessly integrates usability, low-latency real-time insights, and actionable intelligence—empowering both individuals and industries to make timely, well-informed weather-dependent decisions.

6. Conclusion

This work presents the development of a real-time weather forecasting mobile application that effectively integrates IoT sensors and machine learning models to generate accurate and accessible weather predictions, with an initial focus on Gazipur. By incorporating LoRa-based communication, the system overcomes one of the major limitations of IoT networks, the Internet dependency, while enabling long-range, energy-efficient data transmission.

The prototype demonstrates the promise of combining IoT infrastructure and data-driven intelligence to meet localized weather monitoring needs. While currently limited to a specific region, the system establishes a scalable and adaptable foundation that can be extended across the country. With plans to deploy additional weather stations, enhance ML training data, and incorporate user-centric improvements, our project is well-positioned to become a reliable, cost-effective, and widely adopted solution for real-time weather forecasting. Ultimately, this innovation contributes to data-driven decision-making in sectors such as agriculture, disaster preparedness, and public awareness. By refining the system and expanding its capabilities, we aim to ensure its long-term success and wide-scale adoption, ultimately contributing to better, data-driven decision-making in weather forecasting.

7. Challenges and future scope

While our system demonstrates significant promise, a few limitations remain. Currently, the deployment is limited to Gazipur, which restricts the geographical diversity of the training data and may affect model generalizability across different climatic regions. Additionally, although LoRa integration using Arduino and ESP32 modules enables long-range, low-power data transmission without relying on continuous internet connectivity, making it ideal for rural or infrastructure-scarce areas expanding the network across broader regions presents logistical and maintenance challenges.

Looking ahead, we plan to incorporate more advanced machine learning models within our incremental learning framework to further boost prediction accuracy and robustness. We also aim to deploy additional weather stations across diverse locations in Bangladesh, enabling the collection of richer, localized datasets. This will allow our models to adapt more precisely to microclimatic variations, enhancing forecast reliability. Furthermore, by continually gathering real-time weather data, we will build a progressively larger dataset, strengthening the predictive capability of our system over time. Based on user feedback, we also intend to introduce new features within the mobile application to further elevate the user experience and streamline access to localized weather insights.

CRediT authorship contribution statement

Jul Jalal Al-Mamur Sayor: Writing – original draft, Validation, Software, Resources, Project administration, Methodology, Investigation, Formal analysis. **Nishat Tasnim Shishir:** Writing – original draft, Software, Methodology, Investigation, Formal analysis. **Bitta Boibhov Barmon:** Software, Methodology, Investigation. **Sumon Ahemed:** Validation, Data curation. **Md. Moshiur Rahman:** Writing – review & editing, Supervision, Software, Project administration, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The complete code for our project, including the frontend, backend, and database, is available at https://github.com/Software-Machine-Intelligence-Lab/Weather-Station.

References

[1] Agyekum Thomas Peprah, Antwi-Agyei Philip, Dougill Andrew J. The contribution of weather forecast information to agriculture, water, and energy sectors in East and West Africa: A systematic review. Front Environ Sci 2022;10:935696.
[2] Doblas-Reyes F, Garcia Anice, Hansen James, Mariani Luigi, Nain Ajeet, Ramesh Kulasekaran, Venkataraman R. Weather and climate forecasts for agriculture. Guid Agric Meteorol Pr 2003;57.
[3] Gayialis Sotiris P, Kechagias Evripidis P, Konstantakopoulos Grigorios D. A city logistics system for freight transportation: Integrating information technology and operational research. Oper Res 2022;22(5):5953–82.
[4] Steinker Sebastian, Hoberg Kai, Thonemann Ulrich W. The value of weather information for e-commerce operations. Prod Oper Manage 2017;26(10):1854–74.
[5] Buluttan. How weather influences renewable energy production. 2023, URL: https://www.buluttan.com/blog/climate-change/how-weather-influences-renewable-energy-production.
[6] Naveen L, Mohan HS. Atmospheric weather prediction using various machine learning techniques: A survey. In: 2019 3rd international conference on computing methodologies and communication. ICCMC, IEEE; 2019, p. 422–8.
[7] Guanche. Why weather apps fail to accurately predict the forecast and what can be done about it. 2024.
[8] Zhang Guodao, Navimipour Nima Jafari. A comprehensive and systematic review of the IoT-based medical management systems: Applications, techniques, trends and open issues. Sustain Cities Soc 2022;82:103914.
[9] Sayor Jul Jalal Al-Mamur, Shishir Nishat Tasnim, Barmon Bitta Boibhov, Ahemed Sumon, Nurjahan, Whaiduzzaman Md. RoboServe: Design and implementation of a robotic agent for serving food at restaurants. In: Proceedings of the 3rd international conference on computing advancements. 2024, p. 588–97.
[10] Joseph Ferdin Joe John. IoT based weather monitoring system for effective analytics. Int J Eng Adv Technol 2019;8(4):311–5.
[11] Weather monitoring system market. 2024, https://dataintelo.com/report/weather-monitoring-system-market. [Accessed 10 July 2025].
[12] Global weather forecasting services market size, share, and trends analysis report – industry overview and forecast to 2032. 2025, https://www.databridgemarketresearch.com/reports/global-weather-forecasting-services-market. [Accessed 10 July 2025].
[13] Global weather forecasting services market size, share, and trends analysis report – industry overview and forecast to 2032. 2025, https://www.thebusinessresearchcompany.com/report/weather-forecasting-services-global-market-report. [Accessed 10 July 2025].
[14] NiuBol. How much does an automatic weather station cost? 2022, Time: 2022-07-16 10:39:19, Popularity: 2331. URL: https://www.niubol.com/Product-knowledge/weather-station-cost.html.

[15] Yin Hui, Aryani Amir, Lambert Gavin, Wu Zhuochen, Nambiar Nakul, White Marcus, Salvador-Carulla Luis, Sadiq Shazia, Sojli Elvira, Boddy Jennifer, et al. Leveraging artificial intelligence technology for mapping publications to sustainable development goals. Array 2025;100419.
[16] Girija C, Grace Shires Andreanna, Harshalatha H, Pushpalatha HP. Internet of things (IOT) based weather monitoring system. Int J Eng Res Technol (IJERT) 2018.
[17] Kamble SB, Rao P Ramana P, Pingalkar Anurag S, Chayal Ganesh S. IoT based weather monitoring system. Int J Adv Res Innov Ideas Educ 2017;3(2):2886–991.
[18] Rao Bulipe Srinivas, Rao K Srinivasa, Ome N. Internet of things (IoT) based weather monitoring system. Int J Adv Res Comput Commun Eng 2016;5(9):312–9.
[19] Piciullo Luca, Abraham Minu Treesa, Drøsdal Ida Norderhaug, Paulsen Erling Singstad. An operational IoT-based slope stability forecast using a digital twin. Environ Model Softw 2025;183:106228.
[20] Ben Bouallègue Zied, Clare Mariana CA, Magnusson Linus, Gascón Estibaliz, Maier-Gerber Michael, Janoušek Martin, Rodwell Mark, Pinault Florian, Dramsch Jesper S, Lang Simon TK, et al. The rise of data-driven weather forecasting: A first statistical assessment of machine learning–based weather forecasts in an operational-like context. Bull Am Meteorol Soc 2024;105(6):E864–83.
[21] Gotmare Vaishnavi, Kolte Rajesh, Thengodkar Rutwik. Weather monitoring system using Arduino UNO-1. Int Eng J Res Dev (IEJRD) 2021.
[22] Nallakaruppan MK, Kumaran U Senthil. IoT based machine learning techniques for climate predictive analysis. Int J Recent Technol Eng (IJRTE) 2019;5:171–5.
[23] Alam Md Jahirul, Rafi Shoyeb Ahammad, Badhan Ali Adnan, Islam Md Najmul, Shuvo Saiful Islam, Saleque Ahmed Mortuza. Low cost iot based weather station for real-time monitoring. In: 2020 IEEE 2nd international conference on circuits and systems. ICCS, IEEE; 2020, p. 127–33.
[24] Verma Gaurav, Mittal Pranjul, Farheen Shaista. Real time weather prediction system using IOT and machine learning. In: 2020 6th international conference on signal processing and communication. ICSC, IEEE; 2020, p. 322–4.
[25] Kapoor Palak, Barbhuiya Ferdous Ahmed. Cloud based weather station using IoT devices. In: TENCON 2019-2019 IEEE region 10 conference. TENCON, IEEE; 2019, p. 2357–62.
[26] Math Rajinder Kumar M, Dharwadkar Nagaraj V. IoT based low-cost weather station and monitoring system for precision agriculture in India. In: 2018 2nd international conference on I-SMAC (IoT in social, mobile, analytics and cloud)(I-SMAC) I-SMAC (IoT in social, mobile, analytics and cloud)(I-SMAC), 2018 2nd international conference on. IEEE; 2018, p. 81–6.
[27] Bonilla J, Carballo JA, Abad-Alcaraz V, Castilla M, Álvarez JD, Fernández-Reche J. A real-time and modular weather station software architecture based on microservices. Environ Model Softw 2025;106337.
[28] Fowdur Tulsi Pawan, Ibn Rosun Mohammad Nassir-Ud-Diin, et al. A real-time collaborative machine learning based weather forecasting system with multiple predictor locations. Array 2022;14:100153.
[29] Kodali Ravi Kishore, Sahu Archana. An IoT based weather information prototype using WeMos. In: 2016 2nd international conference on contemporary computing and informatics. IC3I, IEEE; 2016, p. 612–6.
[30] Krishna A, et al. IoT and ANN-based weather monitoring system. J IoT Syst 2023;7(2):145–52.
[31] Perakis T, et al. Advancements in large-scale weather monitoring systems using IoT and ML algorithms. IEEE Trans IoT 2024;9(1):35–45.
[32] Satyanarayana KNV, Reddy SRN, Varma KNV Suresh, Raju P Kanaka. Mobile app & iot based smart weather station. Int J Electron Commun Instrum Eng Res Dev (IJECIERD) 2017;7(4):1–8.
[33] Singh S, et al. Weather forecasting using IoT and machine learning techniques. Int J Adv Comput 2022;11(3):120–9.
[34] Meshram Kundan, Mishra Umank, Rathnayake Upaka, et al. Application of artificial intelligence in agri-tech, environmental and biodiversity conservation. Array 2025;100412.
[35] Visual Crossing. Weather Data & API: Global Forecast & History Data. URL: https://www.visualcrossing.com/.
[36] Bolikulov Furkat, Nasimov Rashid, Rashidov Akbar, Akhmedov Farkhod, Young-Im Cho. Effective methods of categorical data encoding for artificial intelligence algorithms. Math 2024;12(16):2553.
[37] Barjatiya Pratik. Unleashing the power of random forest: Why it outperforms decision trees and expert rules. Medium 2024. URL: https://pratikbarjatya.medium.com.
[38] DataCamp. What is incremental learning? DataCamp Blog 2023. URL: https://www.datacamp.com/blog/what-is-incremental-learning.
[39] Trikhatri Silash. A project report on creating back-end for online business site utilizing fastAPI (Ph.D. thesis), SRM UNIVERSITY; 2022.
[40] RoboticsBD Bangladesh. RoboticsBD store. 2024, URL: https://store.roboticsbd.com/.
[41] Jim Frost. Root Mean Square Error (RMSE). Statistics By Jim. URL: https://statisticsbyjim.com/regression/root-mean-square-error-rmse/.
[42] Gupta Hoshin V, Kling Harald, Yilmaz Koray K, Martinez Guillermo F. Decomposition of the mean squared error and NSE performance criteria: Implications for improving hydrological modelling. J Hydrol 2009;377(1–2):80–91.
[43] EUMeTrain. Verification of Continuous Variables. URL: https://resources.eumetrain.org/data/4/451/english/msg/ver_cont_var/uos3/uos3_ko1.htm.
[44] Bolla Sreenivasulu, Anandan R, Thanappan Subash. Weather forecasting method from sensor transmitted data for smart cities using IoT. Comput Intell Neurosci 2022;2022:1–11. http://dx.doi.org/10.1155/2022/5952801.
[45] Abhishek R, Soni A, Mishra R. Weather forecasting using machine learning algorithms. Int J Eng Res Technol (IJERT) 2021;10(5):1–5, URL: https://www.ijert.org/weather-forecasting-using-machine-learning-algorithms.