ECE 445

SENIOR DESIGN LABORATORY

DESIGN DOCUMENT

Machine Learning-based Weather Forecast on Arduino

Team #26

XUANYU CHEN
(xuanyuc2@illinois.edu)
ZHEYU FU
(zheyufu2@illinois.edu)
ZHENTING QI
(qi11@illinois.edu)
CHENZHI YUAN
(chenzhi2@illinois.edu)

Sponsor: Cristoforo Demartino

TA: Yi Wang

May 22, 2023

Abstract

We have developed a machine learning-based weather forecasting system that provides accurate and timely predictions for our surrounding areas. By analyzing data from various sources, including weather sensors and historical records, our system generates real-time weather forecasts. Incorporating advanced techniques such as pattern recognition and artificial neural networks, we have created a comprehensive and accurate weather forecasting model. Also, to make our system more accessible and User-Friendly, we develop a web application that can empower users to make informed decisions based on the most current weather data available. Our system continuously learns from new data, improving prediction accuracy and adapting to changes in weather patterns. Our objective is to overcome limitations of traditional weather stations and provide more reliable, location-specific forecasts. We envision our system as an accessible and user-friendly tool, with a web application that allows users to access weather forecasts on the go and make informed decisions based on up-to-date weather information.

Key words: Arduino, smart weather station, weather forecasting

Contents

1 Introduction

1.1 Background

Weather forecasting has become an indispensable part of people's lives. It deeply influences people's decision-making in so many aspects. Mostly, people routinely resort to publicly published real-time weather data and weather predictions to inform themselves of the temperature, humidity, and whether or not it will rain. However, publicly reported weather data are collected from the weather station, which could be within a long distance, and have a completely different environment from where people work or live. Thus, we design and develop a weather forecasting system that leverages the power of machine learning to produce accurate and timely weather predictions for our surrounding areas.

Our system will analyze vast amounts of data from various sources, including weather sensors and historical weather data, to generate real-time weather predictions. We also incorporate advanced techniques such as pattern recognition and artificial neural networks to create a comprehensive and accurate weather forecasting model. The system can continuously learn from the new data to improve the accuracy of the predictions, making it a self-adaptive system that can handle changes in weather patterns over time.

1.2 Objective

1. Great Weather Forecasting Accuracy and Accessibility
   The accuracy of weather forecasting plays a crucial role in making informed plans and preparations, especially in regions where traditional weather stations may not be available or their predictions are unreliable due to distance and altitude factors. Our primary objective is to build a machine learning-based system that overcomes these limitations and provides more reliable and location-specific weather forecasts for our community. By leveraging advanced algorithms and data analysis techniques, we aim to enhance the accuracy of our predictions and deliver reliable weather information to users.

2. Great Accessibility and User-Friendliness
   By developing an OLED real-time weather data display screen, we aim to cater to users' needs and provide them with intuitive tools to interact with weather data effectively. We also develop a user-friendly web application as part of our system. This application will allow users to access weather forecasts on the go, providing them with up-to-date and real-time weather information. By incorporating user-friendly features and a responsive design, our objective is to empower users to make informed decisions based on the most current weather data available.

3. Tailored Weather Forecasts
   Understanding the diverse needs of users, our objective is to provide tailored weather forecasts that cater to individual preferences and requirements. Through our system, users will have the ability to customize the weather information they receive

based on their specific interests and purposes. Whether it's for personal planning, outdoor activities, or business operations, our goal is to deliver relevant and personalized weather forecasts.

1.3 High-level Requirements

- The weather measurement prototype with sensors should be able to collect the temperature, humidity, barometric pressure, etc., accurately. To be more specific, our collected weather parameters will be compared to real-time data from a meteorological station and should be within a reasonable margin of error. For example, temperature error should be within 5%, humidity error, and pressure error within 10%.
- A machine learning algorithm should be successfully trained to make predictions on the weather conditions: rainy, sunny, thunderstorm, etc. This is a multi-classification problem and we hope our macro-average precision can reach 0.75 or higher.
- Our system can collect and forecast the weather in Haining, in real-time, and/or longer-period forecast. Specifically, our hardware subsystem could collect and upload real-time weather information to the software subsystem every 60 minutes. And our software subsystem could predict the following 4 hours' weather conditions based on the recently inputted 48 hours' weather parameters. For longer-period prediction, our system will be able to predict the weather up to 4 days later, but it could be a great challenge to predict accurately in this case as you can imagine.
- The forecast weather information could be demonstrated elegantly through some UI interface.

1.4 Block Diagram

Figure 1 illustrates our entire design. Our design can be divided into two subsystems.

1. Hardware Subsystem
   Considering the complexity of weather conditions, our system incorporates the following weather indicators and their corresponding collectors: a humidity and temperature sensor, a barometric pressure sensor, a rain sensor, a PM2.5 Air Quality sensor, a UV detector, and an anemometer for wind speed. The aforementioned equipment will be integrated into Arduino, covered with a waterproof enclosure. All sensed indicators will be used for weather prediction and transmitted to our database by our data transmission module. The data transmission module consists of wireless transmission from outdoor Arduino and indoor Arduino and wired serial transmission from indoor to personal computer via serial port. Besides, an OLED display screen showcases the real-time sensed weather data. Our power supply for the weather station consists of batteries, a voltage booster, solar panels, and a solar panel charging circuit.

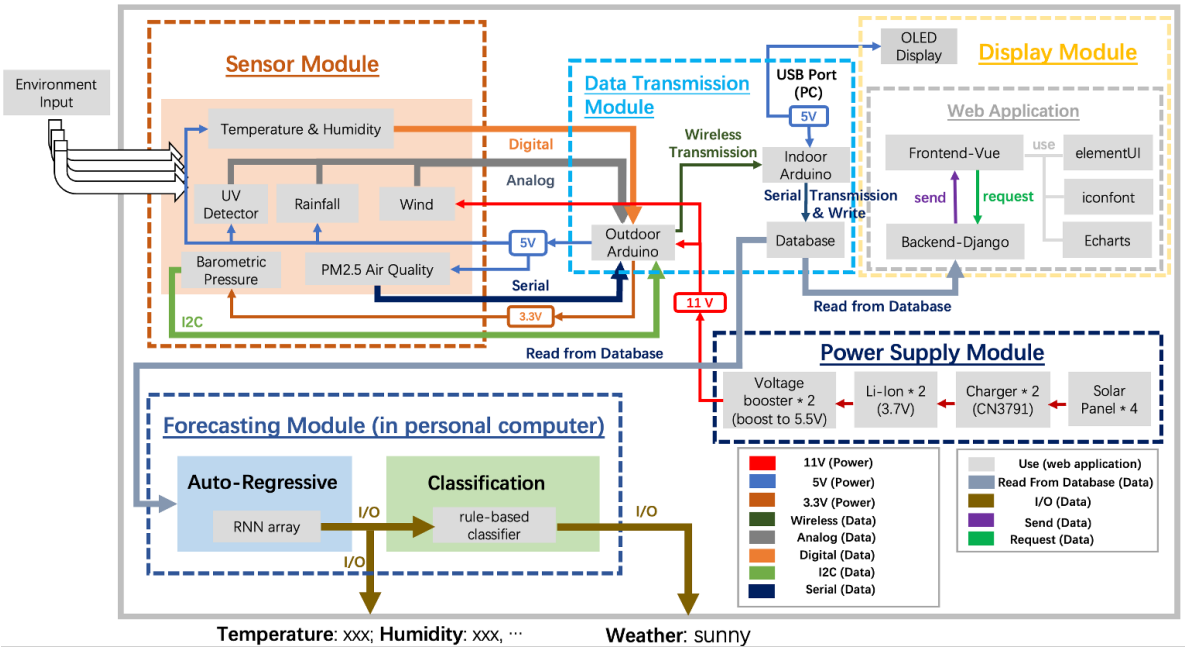


Figure 1: Block Diagram

2. Software Subsystem
   A practically usable weather forecast system is supposed to make reliable predictions for real-world multi-variable weather conditions. We apply machine learning techniques to suffice such generalization to unseen data. To this end, a high-quality dataset for training and evaluating the machine learning model is required, and a specially designed machine learning model would be developed on such a dataset. After a preliminary investigation, we have downloaded Haining's weather datasets for the most recent 40 years from the OpenWeather platform. And we believe auto-regressive models could serve as practical solutions for our model. Once a well-trained machine-learning model is obtained, we will deploy the model on portable devices with easy-to-use APIs.

2 Design

2.1 Weather Station

Our weather station collects real-time local weather data with 9 measurements. In addition, our weather station has strong water resistance and a robust power supply system.

2.1.1 Sensor module

Our sensor module is deployed on Arduino Uno, a microcontroller board based on the ATmega328P. We attach great importance to the sensor selection, ensuring that the mea-

sured types of weather parameters match up with those in our datasets for building our machine-learning models. Additionally, we make sure the selected data types are comprehensive enough to be generalized to an overall weather description. Our measurements include 4 basic parameters to be fed into the forecasting model, i.e. temperature, humidity, barometric pressure, and wind speed. In addition, we quantify other useful indicators, including rainfall, ultraviolet radiation, and particle pollution.

Sensor Selection Detailed decisions for sensors are as follows:

- **Temperature** is definitely one of the most important weather parameters. We calibrate our temperature value to lessen the fluctuations by taking the average of two sensor readings from DHT11 [1] and BMP180 [2]. The two sensors' temperature ranges are 0 to 50 °C and -40 to 85 °C with the same accuracy of ± 2°C. This sensing ranges perfectly match the weather condition in our measured area, Haining.
- **Humidity** is measured by the DHT11 sensor. It's a low-cost, long-term stable combined sensor that allows long-distance signal transmission and precise calibration. A big advantage of DHT11 is its fast response time. Sensor readings are fetched directly from a digital pin at a sampling rate of 4s in our design.
- **Barometric Pressure** is measured by the BMP180 sensor. It can precisely detect real-time changes, which contributes to the high sensibility of our data collection system. Also, the chip consumes less than 1 mA during measurements and only 5 μA when idle. Its low power consumption makes it a good choice for our system. The sensor communicates with the Arduino via I2C serial communication protocol. We calibrate the readings to get the sea-level pressure by the equation $P_0 = \frac{P}{(1-(altitude/44330))^{5.255}}$.
- **Wind Speed** is measured by the JL-FS2 three-cup anemometer [3]. The sensor has great hardness, excellent waterproof, and high precision. The minimum starting wind speed is around 0.4m/s, allowing us to measure the wind speed under most weather conditions in Haining. The value is calibrated by adopting the voltage signal from the analog pin.
- **Rainfall** is represented as a boolean value. The value is determined from the differences between the current analog value and the previously sampled one from the water level sensor [4]. The higher the water level, the larger the output voltage.
- **Ultraviolet Radiation** is measured by the solar UV intensity sensor CJMCU-GUVA-S12SD. We apply a basic filter function to filter photocurrent fluctuations when processing the read analog values.
- **Particle Pollution** is evaluated by the concentrations of fine particles of three sizes, i.e., PM1.0, PM2.5, and PM10. AQI values are calculated for the air quality evaluation.

PCB Design Considering that our main components are sensitive to subtle changes in surrounding conditions, we make careful designs for our PCB layout. We put all the sensor components and the wireless transmission module on the front side of our PCB.

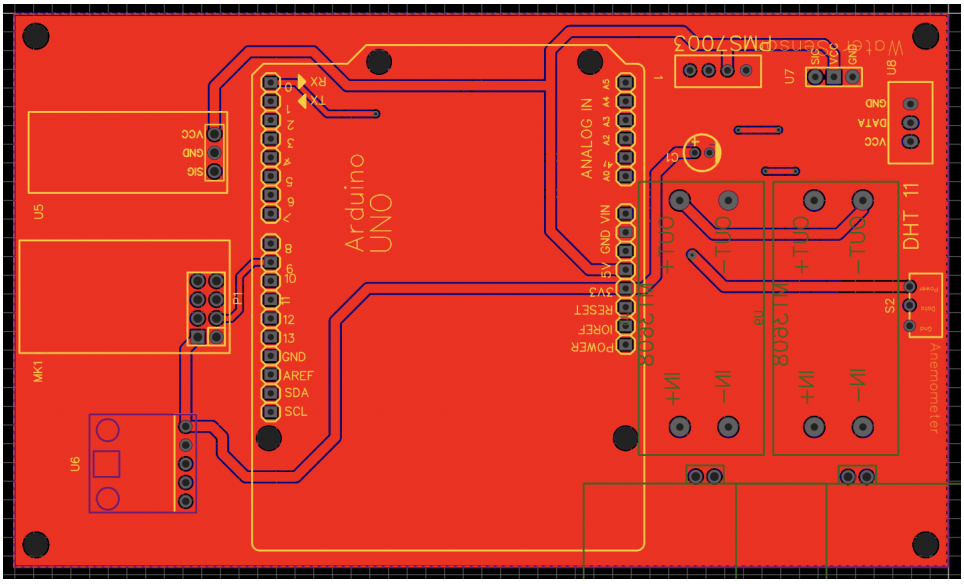


Figure 2: Sensor module PCB design

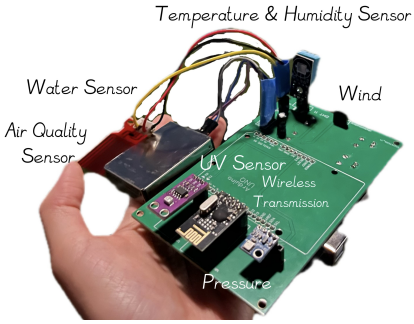


Figure 3: Front side of the sensor PCB

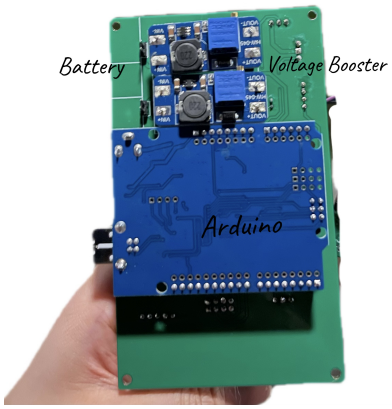


Figure 4: Back side of the sensor PCB

Other components that may generate unexpected heat-ups, including the Arduino board, voltage booster, and battery pins, are placed on the other side. PCB design and physical pictures are shown in Figures 2, 3, 4.

2.1.2 Power Supply Module

For our outdoor weather station, the power source of the Arduino Board is 2 * 3.7 V 4000 mAh batteries with a DC-to-DC Converter Module to boost the voltage from 3.7 V to 5.5 V to finally support stable 11 V (2 * 5.5 V) for our outdoor Arduino and anemometer. Other sensors will be directly powered by outdoor Arduino. Arduino provides three voltage levels, including 3.3 V for the water-level sensor, and 5 V for the remaining components. For the indoor receiver, Arduino will be charged with the USB cable.

To ensure that our system won't suffer from failure because of a dead battery, a solar panel

charging system is designed to ensure the sensor module with the Arduino is powered continuously. The circuit is shown in Figure 5. The Lithium-Ion battery will be charged through a charger, the CN3791, powered by the solar panels to make our system sustainable.

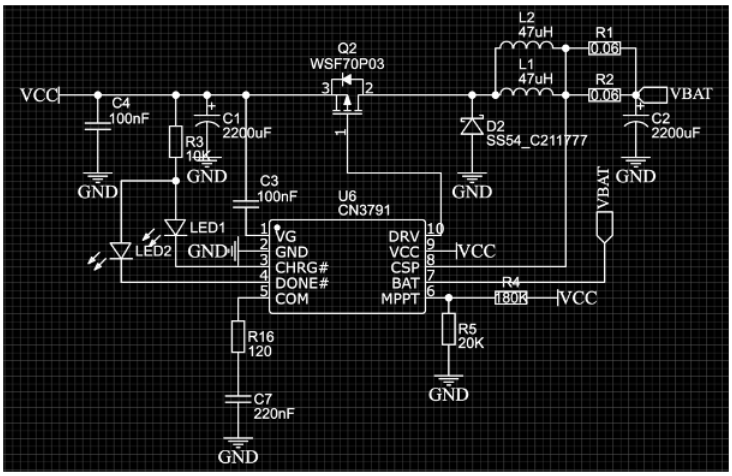


Figure 5: Power supply schematic

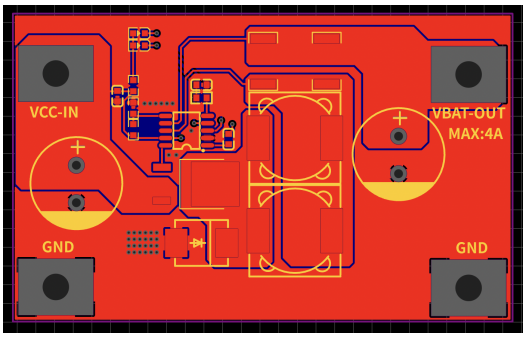


Figure 6: PCB layout of power supply

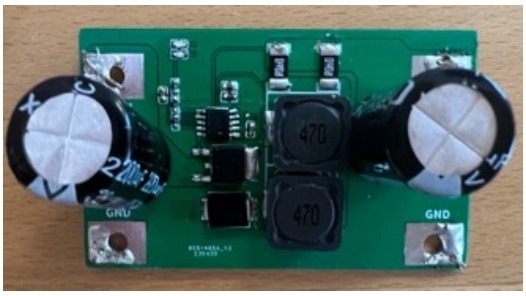


Figure 7: Completed power supply PCB

2.1.3 Mechanical Design

Water-proof Enclosure To protect the circuit, we will also design a highly waterproof shell for our hardware subsystem. Due to such consideration, we designed a double-enclosure structure. The transparent is the outer enclosure, and the red and white ones are the inner enclosures. PCBs would be put inside the white enclosure; the battery is placed in the red box. For requiring measuring Temperature, Air Pressure, Light, Raindrops, Humidity, Wind speed, and direction, we need to carefully consider the distribution of the sensors and PCBs. To keep the components away from water, the enclosure should reach the IP42 level waterproofing, which means the water flow less than 15 degrees to the vertical direction could be prevented from entering the enclosure. The outfit layout is shown below.

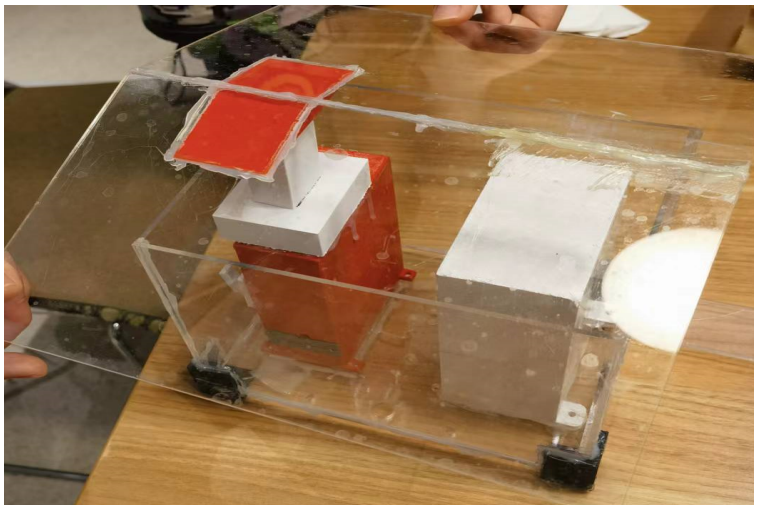


Figure 8: The structure for the enclosure

2.2 Data Transmission

Data transmission in our system consists of three processes: the wireless connection between the outdoor weather station and the indoor receiver, serial communication between the indoor receiver and our personal computer (PC), and database operations on our local database. Instead of directly transmitting the data from the weather station to our PC with additional routers, the two-stage transmission is more economical. It allows data display on an OLED screen in the intermediate stage, providing an easy and direct way to observe refreshing data.

Wireless transmission As Figure 10 demonstrates, we use the low-power NRF24L01 Bluetooth chip for wireless data transmission. The module is able to correctly transmit the data over long distances, at a maximum of 28 m, in our experiment. It also tolerates barriers with several pedestrians walking across our transmission route.

- **Data Packet** The NRF module has a dynamic 32-byte payload length. Therefore, we transmit our pure 40-byte data in two separate packets. The formats are shown in Figure 9. Each packet starts with a 2-byte distinguished header, and the following parameters are separated by a single-byte symbol (commas are used in our design).

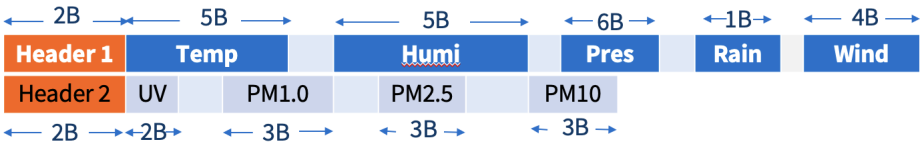


Figure 9: Payload field of the NRF data packet

- **Error Detection** We use the built-in cyclic redundancy check (CRC) in the data communication field to detect accidental errors in wireless transmission. It is worth

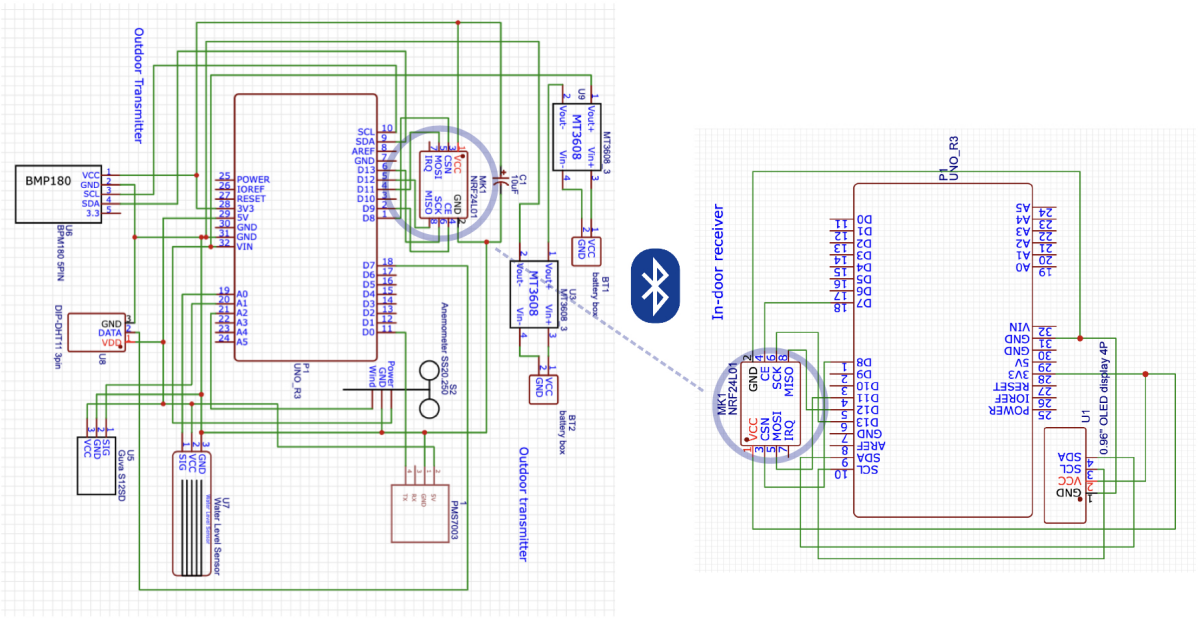


Figure 10: Wireless communication schematic

noticing that the CRC is calculated over the whole packet, including address, PID, and payload. This extra packet acceptance requirement guarantees data integrity.

- **Transmission Rate** We transmit a packet every 2 seconds. Thus, it takes approximately 4 seconds to refresh a complete piece of record, with a slight delay within a second on some occasions.

Serial communication Once the indoor Arduino receives an intact piece of record, it will transmit the data via serial communication to our PC. PySerial packages are used to establish the connection. Once the receiver encounters a linebreak, it will parse the input buffer from a "Start" token, which indicates the beginning of a piece of record. Until all nine parameters are well received, the record will be stored in the local SQLite database with a timestamp. These timestamps will be further utilized for data filtering. The functional process is shown in Figure 11.

Databse Operations We choose SQLite database for its simplicity, portability, and reliability. It works great with our low-traffic data requests. We use the SQLalchemy toolkit to map our self-defined weather data class to the SQLite database. And the SQLite library is used to carry out I/O operations.

2.3 Forecasting Model

Deep Learning (DL) models, an important branch of ML models, have shown their extraordinary ability to discover complex patterns and features of various types of data. In various families of DL methods, auto-regressive models are specifically designed for processing and generating time series. Since our algorithm's goal is to predict the following

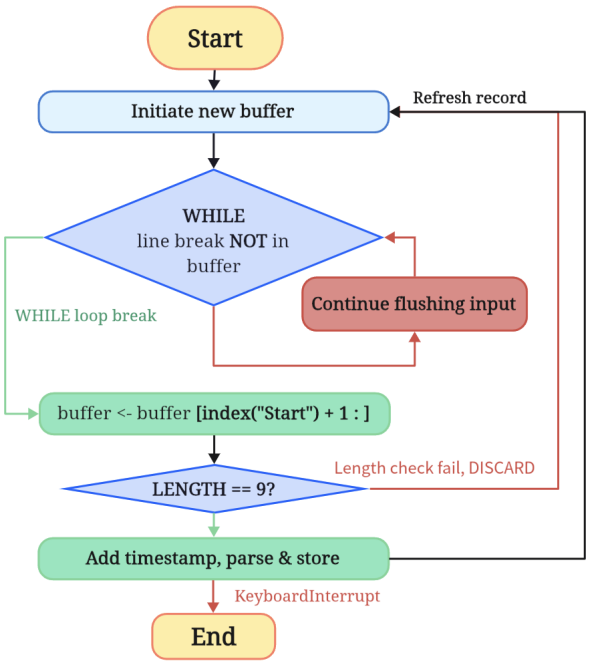


Figure 11: Flowchart of receiver side of serial communication

values in a sequence of values after taking in the current ones, we leverage auto-regressive models' ability to achieve our goal.

LSTM: Long Short-Term Memory We choose LSTM [5] as our base model. LSTM is particularly useful when dealing with sequences of variable length and long-term dependencies between data points. In the context of predicting values in a sequence of weather data, LSTM can be used to learn the underlying patterns in such data and make accurate predictions based on the current input.

The forward propagation of a single LSTM cell at time t is formulated as follows:

$$i_t = \sigma(W_{ii}x_t + b_{ii} + W_{hi}h_{t-1} + b_{hi}) \tag{1}$$
$$f_t = \sigma(W_{if}x_t + b_{if} + W_{hf}h_{t-1} + b_{hf}) \tag{2}$$
$$g_t = tanh(W_{ig}x_t + b_{io} + W_{hg}h_{t-1} + b_{hg}) \tag{3}$$
$$o_t = \sigma(W_{io}x_t + b_{io} + W_{ho}h_{t-1} + b_{ho}) \tag{4}$$
$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \tag{5}$$
$$h_t = o_t \odot tanh(c_t) \tag{6}$$

where x_t is the features of interest (including temperature, pressure, humidity, wind speed, rainfall, and month) that have been normalized, and h_t is the embedding calculated for x_t. LSTM cells are concatenated together into an encoder, after which an MLP (Multi-Layer Perception) is attached to map the hidden states into expected data. The LSTM and the MLP constitute a forecaster, which is trained with an MSE loss function and is expected to predict one future value based on existing ones (in an auto-regressive

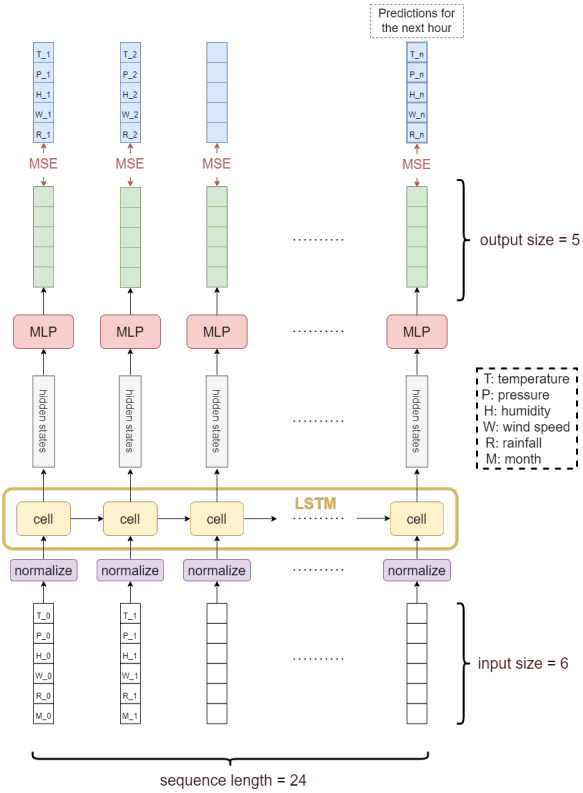


Figure 12: Illustration of the training procedure.

manner). The entire training procedure is illustrated in Figure 12, and is formulated in Algorithm 1.

During training, the LSTM learns to recognize patterns in the data and adjust its parameters to minimize the difference between its predictions and the actual output values. Once the model is trained, we can use it to predict the next values of some weather conditions of interest based on the current input in real-world settings.

Rule-based Classifier A rule-based classifier analyzes data based on a set of pre-defined rules. In the context of weather forecasting, this means creating a set of rules that dictate how different weather variables should be interpreted and combined to arrive at an overall description of the weather.

To create a rule-based classifier for weather forecasting, we would first need to identify the variables that are most relevant for predicting the overall weather conditions. This might include variables such as temperature, pressure, wind speed, humidity, and more. Then, the classifier can be trained on our weather dataset to create complex rules itself. We specifically use Decision Tree to implement the classifier due to its high interpretability and efficiency in computation.

As more rules are established, the accuracy of the classifier should improve, allowing us to provide more accurate and detailed descriptions of the weather. As for why we do not use a Deep Learning model like MLP for such a purpose, the main reason is that such

Algorithm 1 Training a Forecasting Model.

Require: T: number of epochs; S: training dataset; f: LSTM; g: MLP; D: output size
Ensure: Trained model $(f \cdot g)(\cdot)$
  Initialize f and g.
  for epoch t from 0 to T-1 **do**
    for batch b in S **do**
      get sequence data x and label y.
      get LSTM output: $h \leftarrow f(x)$.
      get MLP output: $\hat{y} \leftarrow g(h)$.
      compute MSE loss: $l \leftarrow \frac{1}{D}\sum_{i=0}^{D-1}(y_i - \hat{y}_i)^2$.
      backward propagation from l.
    end for
  end for

models are hard to interpret and understand. With a rule-based classifier, it is easy to see exactly which rules are being applied and how they are being combined to make a prediction. With a Deep Learning model, however, it can be more difficult to understand how the model is making its predictions and what factors are most important for determining the overall description of current weather conditions.

2.4 Web Application

We develop a user-friendly web application to provide a clear and straightforward method to view real-time weather data and predictions.

- **Framework** Django is a powerful Python web framework for building REST APIs, while Vue is a versatile framework for designing user interfaces. Thus, we combine the two in our web development. We also use three other visualization tools, iconfont, Apache Echarts, and ElementUI, for an elegant display.
- **Data Requests** We use the HTTP client library called Axios to make promise-based requests from our browser. The real-time display data are the average values of the records stored in the past three minutes. Prediction information is always pulled from the latest records in the prediction database.
- **User-friendly Designs** We use gauges, progress bars, and concentric rings to vividly show the current condition and the moving trends of the next days and hours. We not only care about the values, we also provide many user-friendly designs. We give descriptions for wind speed and level and provide appropriate actions according to the UV level and the air quality. In addition, we raise warnings in extreme weather conditions. We believe that this kind of easy-to-digest information makes a lot of sense.

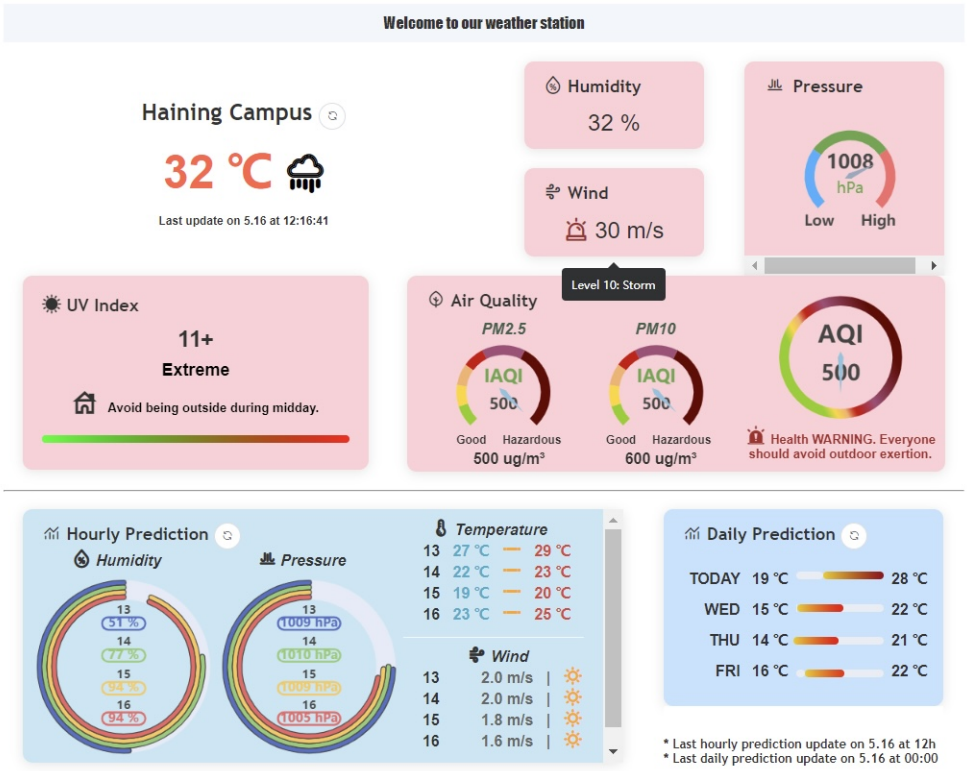


Figure 13: Webpage layout

3 Design Verification

In this section, sensor measurement accuracy, power supply, and forecasting results will be discussed in details. Verifications of the remaining components are described in Appendix A.

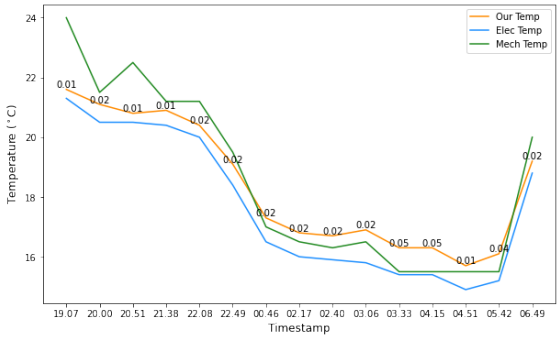
3.1 Sensor Accuracy Analysis

In order to provide accurate real-time weather conditions and forecasts, we pay great attention to the accuracy of our measured data. To reduce fluctuations, we have applied the most suitable calibration methods for every parameter, as mentioned in section 2.1.1. Our system is able to reach the accuracy as listed in table 1.

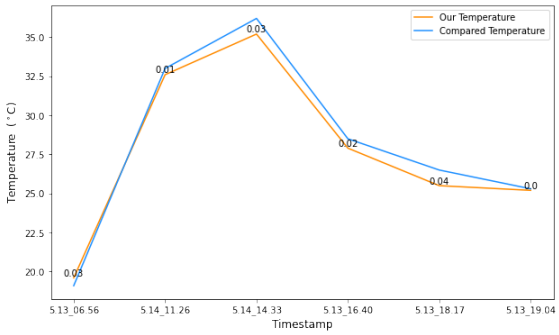
Parameters	Temperature	Humidity	Pressure	Wind speed
Accuracy	± 2.5%	±5%	± 5%	± 5%

Table 1: Sensor accuracy

Our verification methods include a XIAOMI electronic smart sensor (± 1 °C for temperature, ± 4% RH for humidity), a 3-in-1 mechanical sensor (± 2 °C for temperature, ± 8% RH for humidity), Apple's weather App, and a handheld wind sensor.

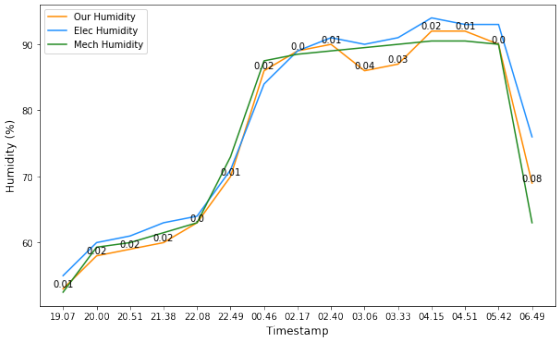


(a) Case 1

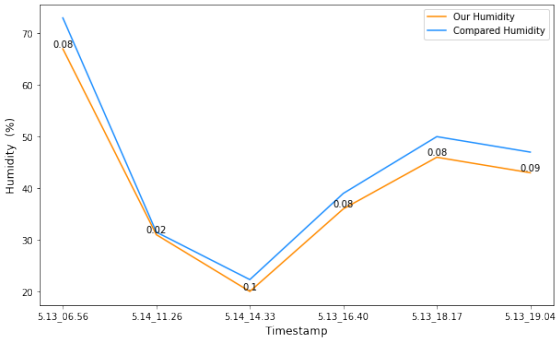


(b) Case 2

Figure 14: Comparisons of temperature readings

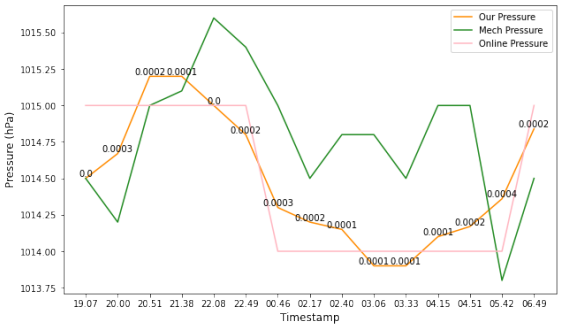


(a) Case 1

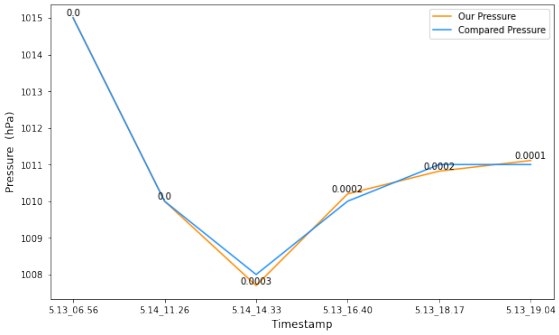


(b) Case 2

Figure 15: Comparisons of humidity readings



(a) Case 1



(b) Case 2

Figure 16: Comparisons of pressure readings

Case 1 In the first case, we recorded the data for a 12-hour period in the evening, approximately one piece of record per hour. We observe a maximum 5% relative difference in temperature, an 8% relative difference in humidity, and almost no differences in pressure. Comparisons are shown in Figures 14a, 15a, 16a.

Case 2 In order to prove that our measurement is accurate throughout the whole day,

we also analyze data at 6 different periods during the daytime. The maximum relative differences we observed were respectively 4% in temperature, 9% in humidity, and tiny differences in pressure again. Comparisons are shown in Figures 14b, 15b, 16b.

Case 3 In the last case, we analyze the wind speed from a 2-minute clip. We observe a long-zero period in Figure 17 since the handheld wind sensor has a minimum start wind speed of around 0.5 m/s. For the remaining period, we observed relatively small differences, and the changing trend is exactly the same.

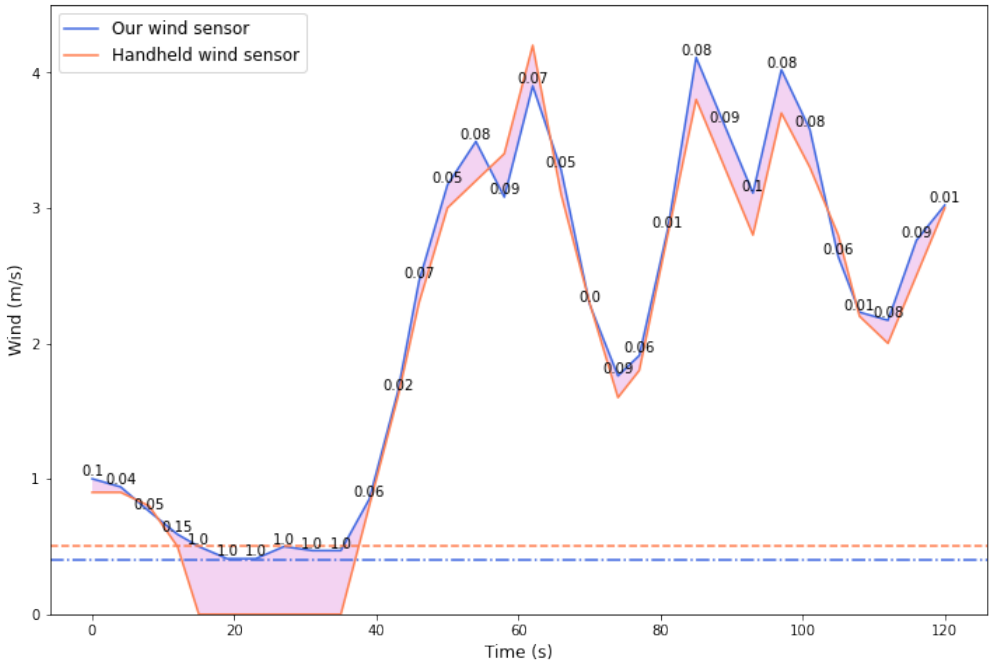


Figure 17: Comparisons of wind speed

3.2 Power Supply Analysis

In our design, the whole power supply module should support the sensing module at a stable voltage that is near 11 V. In the final demo, we measure the voltage difference between VCC and ground pins on our main PCB board and it shows a stable 10.98 V that is very close to 11 V. So this requirement is met.

Another objective for this module is that it should make use of solar energy to make our system sustainable. Firstly, we need to know the working current or working power of our weather station. We did a theoretical analysis of the working power of our weather station as Table 2 shows, which reveals a working power of nearly 1 W, and the working current flowing through each battery to be nearly 135 mA. We also manually test the working current that flows out of each battery, which is nearly 100 mA. Secondly, we test the charging current that flows into each battery and find that the charging current when sunny is about 400 mA with our second version solar charging circuit. Assuming there are 12 hours of sunlight each day, the charging current should be at least 200 mA to make

our system sustainable, and our charging current is enough actually. So this objective is also met.

Components	**Power**
Arduino Uno	Power: $\approx$ 450 mW from Youtube [6]
Wind speed sensor	Power: $\leq$ 300 mW from its data sheet [3]
Water level sensor	Working Voltage: 5 V Working Current: $\leq$ 20 mA Power: $\approx$ 100 mW
DTH11: Temperature and Humidity Module	Working Voltage: 5 V Working Current: 0.3 mA when measuring, 60 μA when standby Power: $\approx$ 0.3 mW
BMP180 Digital pressure sensor	Working Voltage: 3.3 V Working Current: 5 μA Power: $\approx$ 0.02 mW
Light detector sensor	Working Voltage: 5 V Working Current: $\leq$ 5 μA Power: $\approx$ 0.03 mW
nRF24L01 Wireless Module	Working Voltage: 3.3 V Working Current: $\leq$ 13.5 μA Power: $\approx$ 44.55 mW
Other auxiliary resistors	Power: $\leq$ 100 mW

Table 2: Theoretical Working Power Analysis

Conditions	Theoretical WC	Real WC	CC with Version 1 charger	CC with Version 2 charger
Current	around 135 mA	around 100 mA	around 80 mA	around 400 mA

Table 3: Working/Charging current that flows out of/into each battery. WC stands for working current, and CC stands for charging current. Charging current are tested under median strong sunlight on sunny days

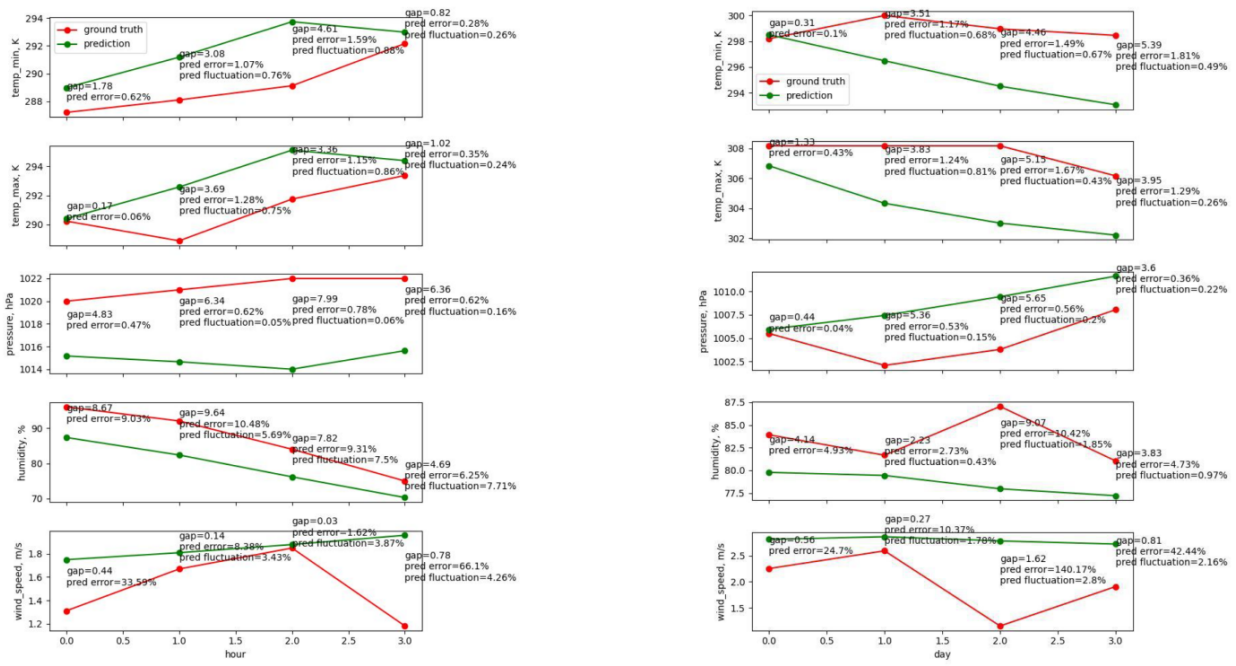


Figure 18: Example Data Batch

3.3 Forecasting Model Analysis

Here we show the plots (Figure 18) of predicted values versus ground truth values (measured by sensors), where green curves denote the predicted values while red curves stand for ground truth values. As we can see from the plots, the trends of the predictions are almost in line with that of the ground truths given four future time steps. Also, the gaps between the two curves are very small. Such agreement in trend also supports the accuracy of our measurements.

Also, we presented detailed statistics of the performance testing of the forecasting model, for both hourly prediction (Figure 19) and daily prediction (Figure 20). We can see that for all these five measures (minimum temperature, maximum temperature, pressure, humidity, and wind speed) and the description category, if we predict by two timesteps, the relative errors, fluctuation rates, and classification accuracies are all within our required range. However, if the model predicts further into the future, we see that the relative errors start to grow significantly, and the predictions for the wind speed even go out of control, reaching a relative error of 45%. We attribute such phenomenon to two main reasons: first, the nature of the auto-regressive model tells us that the more it predicts, the more error it accumulates because it makes new predictions based on its previous predictions; second, wind speed is quite different from the other four measures in that it varies between different locations and different heights, and it depends on many other factors like surrounding buildings and structures. We think that it is impossible for a simple Machine Learning model to capture the changing pattern of such a weather feature. As for the description, which is the output of the classification model, since it takes the former five features as input, the decision path would deviate from the correct one if the input

features are not accurate, leading to wrong predictions.

	Min Temperature	Max Temperature	Pressure	Humidity	Wind Speed	*Description (accuracy)*
Future Hour 1	1.15% / -	0.94% / -	0.33% / -	11.21% / -	23.02% / -	81.83%
Future Hour 2	1.38% / 0.54%	1.44% / 0.54%	0.42% / 0.08%	11.67% / 5.42%	24.84% / 3.84%	79.72%
Future Hour 3	1.32% / 0.93%	1.24% / 0.91%	0.47% / 0.21%	12.73% / 6.1%	33.33% / 4.1%	77.79%
Future Hour 4	1.45% / 0.52%	1.6% / 0.5%	0.48% / 0.17%	14.05% / 5.58%	56.23% / 3.99%	69.53%

*Each cell: relative error / fluctuation rate
*Red: not meet requirement

Figure 19: Hourly Prediction

	Min Temperature	Max Temperature	Pressure	Humidity	Wind Speed	*Description (accuracy)*
Future Day 1	0.6% / -	0.89% / -	0.27% / -	10.75% / -	22.31% / -	80.03%
Future Day 2	1.04% / 0.42%	0.9% / 0.43%	0.31% / 0.11%	11.95% / 2.78%	22.9% / 2.77%	78.21%
Future Day 3	1.18% / 0.44%	1.37% / 0.52%	0.38% / 0.1%	10.71% / 1.51%	45.21% / 3.28%	74.96%
Future Day 4	1.71% / 0.36%	1.3% / 0.35%	0.49% / 0.13%	9.8% / 1.36%	29.39% / 2.03%	71.3%

*Each cell: relative error / fluctuation rate
*Red: not meet requirement

Figure 20: Daily Prediction

4 Costs & Schedule

4.1 Cost

Category	Item	Price
Board	Arduino Uno *2	¥ 300
Display	0.96 OLED screen	¥ 15
Sensors	Temperature & Humidity Sensor (DHT11)	¥ 10
	Barometric Pressure Sensor (BMP180)	¥ 5
	Water Level Sensor	¥ 3
	Adafruit/DFRobot Anemometer	¥ 200-240
	Ultraviolet Radiation Sensor	¥ 15
	Air Quality Sensor	¥ 80
Wireless Communication	nRF24L01 * 2	¥ 10
Power Supply	3.7V Battery *2	¥ 50
	Solar panel *2	¥ 50
	TP4059 Charger	¥ 5
PCB	/	¥ 50
GPU	NVIDIA RTX A5000 * 1	¥ 200
3D Printing Material	Enclosure	¥ 150
Enclosure supporting	/	¥ 100
Weather Station Holder	/	¥ 50
Auxiliary Components	/	¥ 100
Labor Cost	¥$150 * 100 * 4$	¥ 60000
Total		¥ 1440 + *60000 (for labor)*

Table 4: Total Cost

4.2 Schedule

Date	Zhenting Qi	Xuanyu Chen	Zheyu Fu	Chenzhi Yuan
03/20/23	Complete the data preprocessing code.	Complete power supply and wireless communication module schematic & PCB design	Verify sensor selection and complete sensor module PCB design	Design a rough outfit(v1) for the sensors' circuit. Decide the material used in the outfit printing.
03/27/23	Complete the weather forecasting ML model.	Arduino coding for sensors and consult on version 2 sensor schematic	Assemble sensor module (except Anemometer & rain barrel) and complete functionality testings.	Design a rough outfit(v1) for the sensors' circuit.
04/03/23	Complete the code for training and evaluating weather forecasting ML model.	Arduino coding for wireless module and experiments of data transmission.	Assemble power supply module and experiments of functioning duration.	Discuss the waterproofing methods. Add the holes for wire connection on CADs. Verify the Temperature resistance.
04/10/23	Debug all the code and integrate them into a first-version implementation.	Finalize sensor module PCB design including rain barrel and anemometer.	Finalize power and wireless module.	Print the outfit(v2). Modify the outfit CAD model(v3). Verify the UV resistance.
04/17/23	Conduct extensive experiments and analyze the results.	Web application design.	OLED screen display.	Verify the waterproofing method on outfit(v3). Improve the waterproofing method.
04/24/23	Improve the model architecture and implement a last-version system.	Web application design.	Functionality tests for weather station.	Check the PCB waterproofing methods. Verify the final version of the outfit and PCBs assembly.
05/01/23	Prepare final demo.	Prepare design demo testing cases.	Prepare final presentation.	Final check for the assembly.
05/08/23	Mock Demo			

Table 5: Schedule

5 Conclusion

5.1 Accomplishments

With unwavering determination and a collaborative spirit, our team has successfully completed this project that has surpassed all expectations.

One of our key accomplishments was the development of a robust data acquisition module. We designed and implemented a sensor network that collected real-time weather data, including temperature, humidity, wind speed, and atmospheric pressure. These sensor readings served as the foundation for our forecasting models, ensuring that the predictions were based on the most up-to-date and relevant information. Also, we designed a double-layer waterproof shelter to keep our sensors safe and sound.

To facilitate the Machine Learning aspect of our project, we meticulously designed and trained forecasting models using state-of-the-art algorithms. Through a rigorous process of data preprocessing, feature selection, and model training, we developed accurate and efficient prediction models. Our team explored various techniques such as regression analysis, time series forecasting, and ensemble methods to optimize the performance of our models.

Throughout the project, we emphasized the importance of usability and user experience. We developed an intuitive graphical user interface (GUI) that displayed the weather forecasts in a clear and accessible manner. This user-friendly interface allows users to easily interpret the predictions and make informed decisions based on the forecasted weather conditions.

5.2 Uncertainties

One major uncertainty lies in the accuracy of our weather forecasting models. While we have invested considerable effort in training and optimizing our models, there is always the possibility of unforeseen variations or outliers in weather patterns. Factors such as sudden changes in atmospheric conditions, complex local topography, or the emergence of extreme weather events may pose challenges to the accuracy of our predictions. To address this uncertainty, we can consider implementing an ensemble modeling approach, where multiple forecasting models are combined to provide more robust and reliable predictions. By leveraging the collective wisdom of diverse models, we can mitigate the impact of potential inaccuracies in individual models.

Additionally, uncertainties may arise regarding the robustness and reliability of our sensor network. Weather sensors are susceptible to environmental factors such as temperature variations, humidity, and physical damage, which can affect the accuracy and consistency of the collected data. To mitigate these uncertainties, we can implement redundant sensors or periodic calibration procedures to ensure the quality and reliability of the data. Furthermore, regular maintenance and monitoring of the sensor network will be essential to identify and address any issues promptly.

Lastly, uncertainties may arise in the usability and user acceptance of our forecasting system. While we have developed a user-friendly graphical interface, there is always the possibility of users encountering difficulties in interpreting or navigating the system. To mitigate this uncertainty, we can conduct user testing and gather feedback to identify areas of improvement. Incorporating user feedback and making iterative design refinements will help enhance the usability and overall user experience of the system.

5.3 Ethical Considerations

According to the IEEE Code of Ethics 1 [7], we should hold paramount the safety, health, and welfare of the public to strive to comply with ethical design and sustainable development practices. Following this, We are committed to designing our weather forecast equipment to be both practical and ethical. We hope our design can be a next-generation solution for a mini-size meteorological station, and we will try to prevent any form of behavior that may endanger public order with our products.

Also, the IEEE Code of Ethics states that engineers shall maintain confidentiality and protect the privacy of others. We ensure that we use legally available datasets, like Haining's weather conditions from the OpenWeather platform, for training our machine learning models. To avoid ethical breaches, we may implement appropriate measures to safeguard the privacy of data, such as encryption and secure storage.

Moreover, in line with the IEEE Code of Ethics 4 [7], it is important to actively seek, accept, and provide sincere critiques of technical work while acknowledging and rectifying any mistakes. We should be truthful and practical when making statements or predictions based on available data, and give appropriate recognition to the contributions of others.

5.4 Future Work

The prospects of our project lie in its integration and interoperability with other intelligent systems. Currently, our system's output consists of five variables, which we envision as being utilized as inputs for other systems.

Several potential applications of our system have been considered, all of which involve the integration of our system with smart devices. One such scenario pertains to the situation where an individual departs from their residence in the morning and is unexpectedly confronted with rainfall in the afternoon. In this case, our Where the Focus system exhibits a remarkable ability to accurately predict the occurrence of rainfall and autonomously initiate the closure of windows. Distinguishing itself from existing products, which possess the capacity to solely monitor weather conditions and react accordingly, our innovative forecasting system proactively mitigates potential losses by promptly taking appropriate measures.

Alternatively, it is a widely acknowledged fact that larger buildings necessitate a lengthier cooling-down period during the summer season. Within our library, the issue of temper-

ature regulation frequently elicits complaints from patrons, and the response time for adjusting the temperature settings following these complaints is excessively protracted. In addressing this challenge, our system adeptly leverages predicted data to modulate the output power, thereby eradicating the aforementioned delay and ensuring expeditious and efficient temperature control.

References

[1] A. Electronics. "Temperature and Humidity Module DHT11 Product Manual." (2014), [Online]. Available: https://components101.com/sites/default/files/component_datasheet/DHT11-Temperature-Sensor.pdf (visited on 03/07/2023).

[2] B. Sensortec. "BMP180 Digital pressure sensor Datasheet." (2013), [Online]. Available: https://cdn-shop.adafruit.com/datasheets/BST-BMP180-DS000-09.pdf (visited on 03/07/2023).

[3] Q. Electronics. "Anemometer Wind Speed Sensor Datasheet (in Chinese)." (), [Online]. Available: https://cdn-shop.adafruit.com/product-files/1733/C2192_datasheet.pdf (visited on 03/07/2023).

[4] K. Robots. "Water Sensor Module User's Manual." (), [Online]. Available: https://curtocircuito.com.br/datasheet/sensor/nivel_de_agua_analogico.pdf (visited on 03/23/2023).

[5] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[6] M. Klements. "How Long Can An Arduino Run On Batteries? I Tested 6 Of The Most Common Boards." (), [Online]. Available: https://www.youtube.com/watch?v=5cYN5-Spnos (visited on 03/24/2023).

[7] IEEE. "IEEE Code of Ethics." (2016), [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html (visited on 03/07/2023).

Appendix A Requirement and Verification Table

A.1 Sensor Module

Requirements	Verification	Qualification Results
1. The sensor subsystem assembled on the PCB board should be able to measure valid sensor readings.	1. Check if the sensor parameters fall in a reasonable range.	Y
2. Our collected data should reach certain accuracy. We mainly focus on the accuracy of the input parameters of our forecasting model.	2. By calculating the relative differences between readings, we can verify the accuracy requirement. Specific accuracy requirements are mentioned in section 3.1.	Y
3. Sensors should have an acute sense of the changes in the surrounding environment.	3. Observe the changes of readings when • Blow a breath of air at the sensors • Put the hands over the sensors • Cover the water sensor with a wet wipe	Y

Table 6: R&V table for sensor module

A.2 Mechanical Design

Requirements	Verification	Qualification Results
The enclosure of the box should have the high ability prevent the dust and water jet into the outfit. Taking the IP waterproof as a reference, the water flow less than 15 degrees to the vertical direction could be prevented from entering the enclosure.	Vertically dripping water shall have no harmful effect when the enclosure is tilted at an angle of 15° from its normal position. In our assumed test, total of four positions are tested within two axes 2.5 minutes for every direction of tilt, and the water equivalent to 3 mm rainfall per minute.	Y

Table 7: R&V table for mechanical design

A.3 Power Supply Module

Requirements	Verification	Qualification Results
1. The whole power supply module should support the sensing module in a stable voltage that is near 11 V.	1. Measure the voltage difference between vcc and ground pins on our main PCB board.	Y
2. The bare battery (without solar panel so assume no solar energy absorbed) could supply the power for the sensing module over 3V for at least 24-hour duration. It requires our sensing system to have low power cost, and our battery to have enough capacity.	2. Measure the working current flowing out of each battery. Calculate the capacity of the battery divided by the current to get the working duration of each battery.	Y
3. The solar charging circuit should supply 3.7-4.2 V in an almost stable condition to charge the battery.	3. Measure the voltage of the output port of the solar charging circuit.	Y
4. The solar charging circuit could charge the battery with relative high current.	4. Measure the charging current.	Y

Table 8: R&V table for power supply module

A.4 Forecasting Model

Requirements	Verification	Qualification Results
Predicted values for each weather indicator should deviate from true values in an acceptable range. Data predicted would be compared with 1) previously measured data and 2) online weather forecasts. We expect a 25% and 50% relative difference from online values and the last measured values, respectively, for all five parameters.	The predicted value $\hat{y}$ and the true value y (measured at the next time step) for the next time step should satisfy $\frac{\lvert y-\hat{y}\rvert}{\lvert y\rvert} \leq 25\%$ (after normalization). The adjacent predictions should satisfy $\frac{\lvert y_t-y_{t-1}\rvert}{\lvert y_{t-1}\rvert} \leq 50\%$. If this threshold has been exceeded for a certain amount of time steps, we would be informed by the system, and some improvements may be needed.	Y

Table 9: R&V table for forecasting model

A.5 Display Module

Requirements	Verification	Qualification Results
1. The OLED screen should be able to display real-time collected data.	1. Check the screen display. • If the values are reasonable • If the screen refreshes around every 2 seconds. Records are set to be refreshed every 4 seconds and a piece of record will be separately shown on two pages, approximately 2 seconds per page.	Y

Table 10: R&V table for display module

A.6 Data Transmission Module

Requirements	Verification	Qualification Results
1. Our wireless transmission module should be able to transmit the data correctly without any data loss at a distance to be at least 6 m. (The wireless transmission is between outdoor Arduino and indoor Arduino) .	1. Keep the outdoor Arduino at least 6m away from the indoor Arduino. Then we can verify the requirement by checking the received data on the display screen. If all data types are received and the real-time data keeps rolling on the OLED screen, then the requirement is met.	Y
2. Our sensed weather data could be successfully transmitted from the indoor Arduino to our personal computer. (It is wire transmission by serial port)	2. If data received and stored in our database is the same (will also be printed in the terminal) as shown on the OLED, then the requirement is met.	Y
3. Real-time data can be retrieved from our SQLite database and then fed into the forecast model and the web application according to our requirements.	3.1 Data requested by the backend of our web application is the average of records stored in the past 3 minutes. Check the recorded timestamp of those filtered data. 3.2 Data will be read into the prediction model once stored in our database (i.e. new line inserted). Check if the id of the new record is increased by 1 from the former record fed into the model.	Y

Table 11: R&V table for data transmission module

A.7 Web Application

Requirements	Verification	Qualification Results
1. Data can be sent from backend to frontend successfully.	1. Check the GET requests received by the backend, the data sent from backend, and the display on the web page.	Y
2. Elegant display of real-time and predicted data.	2. Check the webpage design.	Y
3. Warnings are raised under extreme weather conditions. Build a more user-friendly weather UI.	3 Check if warning is raised / action provided in the following cases: • Strong gale • Rainy day • Extreme ultraviolet • Hazardous air quality	Y

Table 12: R&V table for web application module