National College of Ireland

IoT Sensor Data vs. Open Data Platforms: A Study on Weather Nowcasting Accuracy

MSc Research Project
MSc in Data Analytics

Henry Egbulam
Student ID: x23266007

School of Computing
National College of Ireland

Supervisor: Mohammed Hasanuzzaman

National College of Ireland

MSc Project Submission Sheet

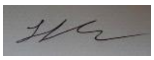
School of Computing

Student Name:	Henry Egbulam
Student ID:	x23266007
Programme:	MSc in Data Analytics **Year:** 2026
Module:	MSc Research Project
Supervisor:	Mohammed Hasanuzzaman
Submission Due Date:	28/01/2026
Project Title:	IoT Sensor Data vs. Open Data Platforms: A Study on Weather Nowcasting Accuracy
Word Count:	6257 **Page Count** 19

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:

Recoverable Signature

X 

Henry Egbulam

Signed by: Henry Egbulam Signature

Date: 27/01/2026

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

IoT Sensor Data vs. Open Data Platforms: A Study on Weather Nowcasting Accuracy

Henry Egbulam
x23266007

Abstract

Accurate hyper-local weather nowcasting is essential for applications ranging from smart home automation systems to outdoor activity planning. However, the reliance on popular open data platforms often fails to capture site-specific microclimatic variations. This study investigates what is termed as the "nowcasting edge" gained by integrating data from a low-cost, personal IoT weather station with standard API data. In order to test this, a robust data pipeline was implemented to synchronise high-frequency sensor readings of temperature, humidity, and pressure collected via a Raspberry Pi-based weather station with data polled from the OpenWeatherMap API. A Stacked GRU neural network was chosen as the optimal architecture for this time-series prediction task. A comparative experiment was conducted between a model trained with IoT-enhanced data and a baseline model relying solely on API data. The results produced confirm the hypothesis to be true. The IoT-enhanced model significantly outperformed the baseline API only model. The RMSE was reduced for temperature by 79.21% and pressure by 90.99% at the 5-minute horizon. While the advantage of the IoT-enhanced model reduces over the forecast window, it still retained a significant 30-83% performance lead at the 60-minute mark. These findings demonstrate that hyper-local sensor data is a critical component for high-accuracy weather nowcasting.

1 Introduction

1.1 Background

The expansion of the Internet of Things (IoT) as a communication model to connect devices such as home appliances, monitoring sensors, and machines to the Internet has led to opportunities for applications to make use of the vast amount of data now being collected (Zanella *et al.*, 2014). These solutions can be applied to a multitude of domains, from home automation to industrial processes, intelligent energy management, and smart grids. Home automation systems are a fast-growing area of interest in the IoT space due to the low-power communication network, such as WiFi, found in most modern homes (Pavithra and Balakrishnan, 2015). An area that has not received as much attention is the monitoring of environmental parameters outside the home.

An IoT-based weather station works by providing information on the changes in the environment in nearby surrounding area. Through the use of low-cost sensors, temperature, humidity and barometric pressure data can be collected and monitored (Kodali and Mandal,

2016). These advantages of having this data available include the ability to notice, or even be notified of, the environmental parameters outside while still being inside the home. It can serve as a reminder to wear warm clothes before heading or to bring in the washing up that is hanging out to dry. While monitoring current conditions is useful, the ability to predict these hyper-local changes in the immediate future would significantly enhance the utility of these smart home systems.

An advanced use of this data is in the realm of predictive analytics. Predictive analytics refers to the use of models developed to forecast future events through the analysis of current and historical data (Kumar and L., 2018). These models are based on statistical and/or machine learning techniques such as regression analysis, artificial neural networks, and support vector machines. Machine learning advances, particularly in the area of time-series forecasting, have allowed for non-statistical experts to calculate predictions from data that has been collected from IoT sensors (Masini *et al.*, 2023).

Attempts to forecast environmental sensor data have seen purchase in the area of weather nowcasting of a hyperlocal area. Nowcasting refers to the short-term forecasting of weather, typically defined as the period from the present up to six hours into the future (Yao *et al.*, 2022). However, there is limited empirical evidence showing that the forecasts calculated from personal IoT sensor data are more accurate than the weather predictions that can be directly obtained or computed from publicly available data provided by online open platforms. Evidence supporting this would promote the use of personal weather stations. The results of this would be of use to industry and hobbyists, such as the agriculture sector or outdoor event organisers.

1.2 Importance

Though previous research has examined the use of IoT sensors for weather nowcasting, it is difficult to find published evidence that these predictions are of higher accuracy than what can be obtained from open data platforms. The objective of this study is to show that data from an IoT-based weather station provides more accurate hyper-localised forecasts than an online weather platform. To test this hypothesis, time-stamped environmental data from both sources will be integrated into a database. A machine learning model will be utilised to compute predictions for the immediate future (up to one hour). The accuracy of predictions from both the IoT data, as well as the publicly available data, will be compared.

1.3 Research question and objectives

The main question this study hinges on is "Does integrating data from personal IoT devices improve the accuracy of localised weather nowcasting compared to predictions derived exclusively from open data platforms?"

To answer this question, this research study follows a procedural set of technical and analytical steps. The first step is to design and implement a robust data pipeline for the ingestion,

synchronisation, and storage of high-frequency weather data from two distinct sources: a local IoT sensor weather station and the OpenWeatherMap API. Building on this initial system, the study seeks to develop and optimise a machine learning architecture suitable for predicting the future values of high frequency time-series environmental variables. As stated in the question, the central objective is to evaluate the existence of a "nowcasting edge" by directly comparing the predictive performance of a model trained with IoT-enhanced data against a baseline model trained only on the open platform data. Finally, the study intends to evaluate the decline of this predictive advantage over a short-term forecast horizon of 5 to 60 minutes to determine the limits of the IoT data's utility.

1.4 Limitations

As this is a small-scale project, there are a number of limitations to be considered. The environmental measurements to be collected from the low-cost IoT weather station are the temperature, humidity, and pressure. This omits several key weather conditions, such as rain and wind speed. Ideally, the environmental sensor data would be collected from various locations to have a more diverse set of conditions, however for this project the data will be measured outside the author’s residence due to accessibility issues.

1.5 Structure of the report

This study includes the following main sections: Introduction, Related Work, Research Methodology, Design Specification and Implementation, Evaluation, and Conclusions. This introductory section describes the basis of the project. The literary review provides a detailed examination of the work related to the forecasting of environmental conditions using machine learning techniques; the papers are grouped into different subsections, each with a main focus area. In the following section, the methodology utilised for integrating the data and calculating predictions is described. The design specification and implementation build on the methodology by addressing the integral parts of the data pipeline architecture and machine learning tools used in development. The result of the study is then presented, utilising Root Mean Squared Error (RMSE) to quantify the comparative performance of the models. Any conclusions reached and future work required are expounded on in the final section of this paper.

2 Related Work

This section explores the current research study in terms of what has already been produced in the broader academic literature. Advancements in three key areas related to this work are examined in detail: IoT sensor-based weather station, the use of open data platforms APIs for predictive analytics, and the performance of machine learning techniques for high-frequency time-series forecasting.

2.1 Weather Nowcasting using IoT Sensor Networks

The increase of low-cost IoT hardware has revolutionised environmental monitoring; IoT sensor networks are now deployable in areas which may have been too deficient of resources to have previously allowed for it. Mohapatra and Subudhi (2022) took advantage of this evolution by developing an inexpensive IoT-based weather monitoring system aimed at enabling real-time environmental data collection and transmission. The system was built using a NodeMCU ESP8266 microcontroller connected to a range of sensors for measuring temperature, humidity, pressure, and rainfall. The sensor data was transmitted via the MQTT protocol to a cloud platform, where it could be stored, visualised, and analysed through a web interface.

The authors prioritised affordability to ensure that the system could be used in resource-constrained environments, such as rural areas, or utilised by the average consumer without hidden software costs. The evaluation of their results showed that they had developed a reliable consumer product with accurate sensor data and reliable communication between the weather station and the cloud platform. It demonstrates the ability of IoT sensor networks to contribute to real-time weather nowcasting by offering localised, high-frequency data streams that can complement or enhance traditional weather forecasting services. In another similar study, Verma *et al.* (2020) utilised the same microcontroller to build a real-time IoT-based weather prediction system. Data was sent to the ThingSpeak cloud for storage, analysis, and visualisation. A logistic regression model was then trained in a Python/Jupyter environment using pre-recorded sensor values. The trained model was applied to real-time inputs to classify weather conditions, with the outcome signalled via an LED on the NodeMCU board. The results showed that this system achieved a high accuracy of approximately 84% and was compared favourably with other approaches in the literature. Together, these studies establish that low-cost hardware can reliably capture high-frequency environmental data.

A common limitation in these static systems is their fixed spatial coverage. Hafeez *et al.* (2021) addressed this by developing a novel weather monitoring system that combined IoT sensors with a mobile robotic platform. Unlike static sensor stations, their design mounted temperature and humidity sensors on a remotely controlled robotic car in order to enable data collection across different areas and environments. The collected sensor data were processed and using a recurrent neural network (RNN) model, short-term weather variables were predicted. Results showed that the RNN produced highly accurate forecasts of indoor and outdoor temperature and humidity trends. To enhance usability, the team also developed an Android application to visualise both real-time sensor readings and predicted values. While these studies prove that IoT systems can collect and predict weather data, they lack a direct, rigorous comparison against established meteorological standards or open data platforms to quantify the improvement of forecasts from using these local sensors.

2.2 Predictive Analytics with Open Weather Data

Parallel to the rise of IoT, the availability of weather data through Application Programming Interfaces (APIs) has enabled scalable predictive modelling without the need for physical hardware. Du *et al.* (2018) explored the development of a REST API designed to provide site-specific historical and near-future weather data in EPW format, specifically for the building simulation community. The researchers were motivated by the need for real-time, accurate weather data, as traditional datasets are often insufficient for predictive building control and performance modelling. Their data pipeline consists of a REST API framework connecting to multiple online forecasts, including OpenWeatherMap, Met.no, and Weatherbit. A number of key environmental parameters were ingested, such as temperature, relative humidity, wind speed, and wind direction. Within their framework, requests can be made using city name, city ID, or geographic coordinates, enabling forecasts at a relatively hyperlocal scale.

In order to validate the approach, forecasts from four API providers were compared with observations from two weather stations: one local station at the University of Navarra and a nearby Spanish National Weather Agency station. The results showed that the API data produced a difference in temperature of 1.6 to 2.2 °C and a difference in relative humidity of 10 to 13% compared to local observations, while the discrepancy between the two observation stations themselves was 0.7 °C and 7%. These results support the hypothesis that while APIs are scalable, they may lack the hyper-local precision required for specific applications.

In a very recent study, Kolambe *et al.* (2025) further demonstrated the use of API-driven systems in weather forecasting. Their application leverages the OpenWeatherMap API with Django's Model–View–Controller (MVC) framework. The system is designed to provide real-time weather updates, forecasts, as well as severe weather alerts. Accessibility is prioritised through features such as multilingual support, unit conversion, and a GPS-enabled location detector.

Interestingly, the literature review of Kolambe *et al.* (2025) references prior work that used low-cost sensors such as DHT11 for weather data collection. The authors do not explain their decision to only rely on API-driven data rather than integrating an IoT device with connected sensors for their solution. This leaves their system vulnerable to the same hyper-local precision issues identified by Du *et al.* (2018). However, their approach does emphasise the benefits of scalability and ease of deployment that systems utilising API data offer. The application integrates multiple online APIs, including OpenWeatherMap, IMD, NOAA, and WeatherAPI, to ensure reliability and support different geographic areas.

The results highlight the system's effectiveness in delivering accurate and responsive weather information with minimal latency (typically under two seconds) and strong cross-platform compatibility. A notable outcome is its ability to notify users of severe weather conditions. This feature would be particularly valuable for agriculture and other outdoor industries. The modular MVC architecture, supported by PostgreSQL for data management and caching mechanisms for efficiency, ensures scalability and maintainability. The authors suggest that future enhancements could include AI-based forecasting models, hyper-local predictions, and personalised weather insights to further improve accuracy and user engagement.

2.3 Machine Learning Architectures for Time-Series Forecasting

The selection of an appropriate forecasting model is critical for nowcasting accuracy. Yamak *et al.* (2020) conducted experiments to compare three of the dominant architectures: Auto-Regressive Integrated Moving Average (ARIMA), Long Short-Term Memory (LSTM), and Gated Recurrent Unit (GRU). The components that make up the linear model, ARIMA, are described as autoregression, differencing, and moving averages. To give a brief description of these terms, autoregression refers to using past values to predict future ones, differencing ensures stationarity, and moving averages filter out short-term volatility. Though RNNs are effective for linear trends, they are often preferred for non-linear data, due to suffering from the "vanishing gradient" problem. The researchers highlight how LSTM networks address this using a complex three-gate structure (input, forget, output) to retain long-term dependencies. Alternatively, the GRU architecture offers a streamlined approach, utilising only two gates (update and reset) to manage information flow. This simpler structure allows GRUs to be computationally more efficient while still handling sequential dependencies. In their empirical tests on cryptocurrency data, Yamak *et al.* found that while ARIMA achieved the lowest error for that specific linear dataset, GRU outperformed LSTM in both accuracy (MAPE of 3.97% vs. LSTM) and training efficiency.

This trade-off between complexity and efficiency was further explored by Im *et al.* (2024). They proposed a machine learning operations (MLOps)-centric data pipeline for energy consumption management, a domain sharing the high-frequency characteristics of weather data. Their system was designed to process real-time data collected from IoT sensors. The pipeline integrates several components (Kafka, InfluxDB, Telegraf, Zookeeper, and Grafana) to support seamless ingestion, storage, visualisation, and prediction. For forecasting, the authors compared seasonal autoregressive integrated moving average (SARIMA) with LSTM models. Their results showed that while SARIMA achieved significantly faster training times (∼1 min 19 s), the predictive accuracy was lower. The LSTM model required more time to train (∼40 min) yet outperformed SARIMA in accuracy, with a notably lower RMSE and Mean Absolute Error.

The authors also validated the efficiency of their data pipeline by optimising Telegraf's configuration. The system achieved average end-to-end processing times of 0.39 seconds for 10,000 records and 1.26 seconds for 100,000 records. These results represent a 30.69–90.88× improvement over a Python-based approach. The pipeline also demonstrated scalability, as the overhead growth was reduced by a factor of 3.07 as data volume increased. These findings highlight the pipeline's suitability for real-time time series applications since it not only focuses on successful predictive modelling, but also efficient data handling.

In the paper 'An Improved Deep Learning Model for High-Impact Weather Nowcasting' by Yao *et al.* (2022), the authors introduced a self-attention–based gated recurrent unit (SaGRU) to improve the prediction of high-impact weather events such as landfalling hurricanes and extreme convective precipitation. Unlike conventional GRU models, SaGRU incorporates a self-attention mechanism that enables it to capture global spatial dependencies within a single

layer, while also enhancing its ability to represent complex atmospheric patterns. The results demonstrated that the SaGRU outperformed traditional GRU and LSTM models. However, the researcher's paper only focused on large-scale radar data. It still remains an open question whether such complex attention mechanisms provide a similar benefit for single-point, hyper-local sensor data, or if the simpler Stacked GRU architecture is sufficient.

2.4 Summary and Research Gap

The literature confirms the feasibility of low-cost IoT weather stations and the utility of open data APIs. It also identifies GRU-based architectures as a promising tool for time-series forecasting. However, there is a notable lack of direct comparative studies that quantify the accuracy gained by integrating these two data sources. Most research focuses either on building the IoT system or on utilising API data. No paper was identified in which one data source was benchmarked against another. This study addresses this gap by directly comparing the predictive performance of an IoT-enhanced model against an API-only baseline, utilising a rigorous deep learning approach.

3 Research Methodology

This section outlines the systematic approach employed to investigate the research question. The methodology was designed as a quantitative, experimental study focused on the comparative analysis of predictive accuracy.

3.1 Data Collection

To address the research question, environmental data was collected from two distinct sources: an IoT device and a publicly available open data platform. For the IoT device, a Raspberry Pi 5 was equipped with an AHT20 sensor to measure temperature and humidity and a BMP280 sensor to measure barometric pressure. As both the sensors are factory calibrated and tested before shipping, no further calibration validation testing was conducted prior to the data collection. This personal weather station was situated outside the researcher's residence to capture the hyper-local conditions. Data was collected at high frequency. It was transmitted every second via the UDP protocol to a local InfluxDB time-series database.

The second data source integrated was the OpenWeatherMap API. A custom Python script was developed to poll real-time weather data using the One Call API 3.0 for the same geographic coordinates as the IoT station. This ensured a direct and valid basis for comparison. The API provided values for temperature, humidity, and pressure. These data points were collected at five-minute intervals and stored in the same InfluxDB database in order to create a centralised repository for both sources.

3.2 Data Preprocessing and Exploration

Following the raw data extraction, the variables needed to be preprocessed before being input in the machine learning model development. A number of Python methods were utilised to transform the data into a clean, consistent, and model-ready format. This phase ensured that the two heterogeneous datasets were now aligned.

First, an initial data cleaning and alignment procedure was performed. To ensure consistency, the relevant data columns from both the IoT and OpenWeatherMap sources were selected and renamed to a uniform schema: temperature, humidity, and pressure. Both datasets were then resampled to a common five-minute frequency using mean aggregation. This was performed so the IoT data matched the polling interval of the OpenWeatherMap API and were therefore temporally aligned. A linear interpolation strategy was applied to handle missing values from intermittent sensor readings or failed API calls. However, this was limited to filling in a maximum of one hour, or 12 five-minute data points to prevent the fabrication of data across larger, multi-day gaps.

After the data was cleaned and aligned, an exploratory data analysis was conducted using Grafana to visualise the time-series and compare trends between the IoT and OpenWeatherMap data. It was also decided that adding time-based features could help the model learn cyclical patterns. These two new features were named, self-explanatorily, as hour_of_day and day_of_week. They derived from the timestamp of each data point.

The final preparatory step for the GRU neural network was to apply data scaling. All numeric features were normalised to a range of [0, 1] using Min-Max scaling. To prevent data leakage and ensure a valid model evaluation, the scaler was fit exclusively on the training portion of the dataset and then used to transform the validation and test sets.

3.3 Machine Learning Model Development

The final phase of the methodology was to develop and evaluate a model capable of addressing the core research question. This process was structured as a multivariate, multi-step time-series forecasting task.

3.3.1 Data Structuring

The pre-processed time-series data was transformed into supervised learning sequences using a sliding window approach. A lookback window of 12 time steps (one hour) was used as the input to predict a forecast window of 12 future time steps. This "sequence-to-sequence" structure was designed to provide the model with sufficient historical context to perform a meaningful one-hour nowcast.

3.3.2 Model Selection and Architecture

Using information gathered from the literature review, the model selection process was reduced to two options. The primary candidate, a Stacked GRU model, was chosen due to its proven effectiveness in capturing sequential patterns and temporal dependencies. This architecture

consisted of two stacked GRU layers of 150 and 75 units, followed by Dropout layers with a rate of 0.15 for regularisation.

This Stacked GRU was then benchmarked against a similar but more complex architecture based on the Self-Attention (SaGRU) mechanism, as proposed by Yao *et al.* (2022) for radar nowcasting. This advanced model, implemented using a Multi-Head Attention layer, was tested to determine if its ability to capture long-range, non-sequential dependencies would provide a performance advantage.

After rigorous testing, the Stacked GRU was definitively selected as the champion model. Even when tuning the learning parameters, it consistently outperformed the more complex SaGRU model, delivering a significantly lower validation loss and more accurate predictions across all forecast horizons. This finding suggests the hyper-local weather signal is dominated by strong, direct sequential patterns that the simpler GRU architecture is better suited to learn.

The final model was compiled using the Adam optimiser with a tuned learning rate of 0.0005 and the mean squared error loss function. To prevent overfitting and ensure the selection of the best-performing model, an EarlyStopping callback was implemented. This caused training to stop when the validation loss no longer showed any improvement.

3.3.3 Evaluation and Hypothesis Testing

To isolate the predictive value of the local sensor data, a controlled experiment was designed. Two identical instances of the champion Stacked GRU model were trained. Model A (IoT-Enhanced) was trained on the complete feature set. This included all data from the IoT sensors, the OpenWeatherMap API, and the time-based features. In contrast, Model B (OWM-Only/Baseline) was trained exclusively on the API data and the time-based features.

The performance of both models was then compared using the RMSE, calculated in the original, unscaled units (C, %RH, hPa) for a simpler understanding. This comparison was performed at three distinct points in the forecast horizon, at 5 minutes, 30 minutes, and 60 minutes ahead, in order to quantify the "nowcasting edge" provided by the local sensor data.

4 Design Specification and Implementation

This section details the technical architecture and implementation strategies used to construct the end-to-end data pipeline, from the physical hardware layer to the final machine learning environment.

4.1 System Architecture

The project was built on a custom data pipeline designed to collect, store, process, and analyse time-series data from two distinct sources. The architectural flow, illustrated in Figure 1, consists of the following key stages:

1. **Data Acquisition:** A hardware-based IoT Weather Station captures hyper-local sensor readings, while a Python script simultaneously polls comparative data from the OpenWeatherMap API.
2. **Ingestion:** Both data streams are transmitted in real-time to a central ingestion point.
3. **Storage:** An InfluxDB time-series database serves as the persistent repository for both raw and processed data.
4. **Processing:** A dedicated ETL (Extract, Transform, Load) script cleanses raw data, performs feature engineering, and writes the model-ready dataset to a separate bucket.
5. **Analytics & Monitoring:** Grafana provides real-time dashboarding, while the Model Development environment consumes the processed data for training and evaluation.

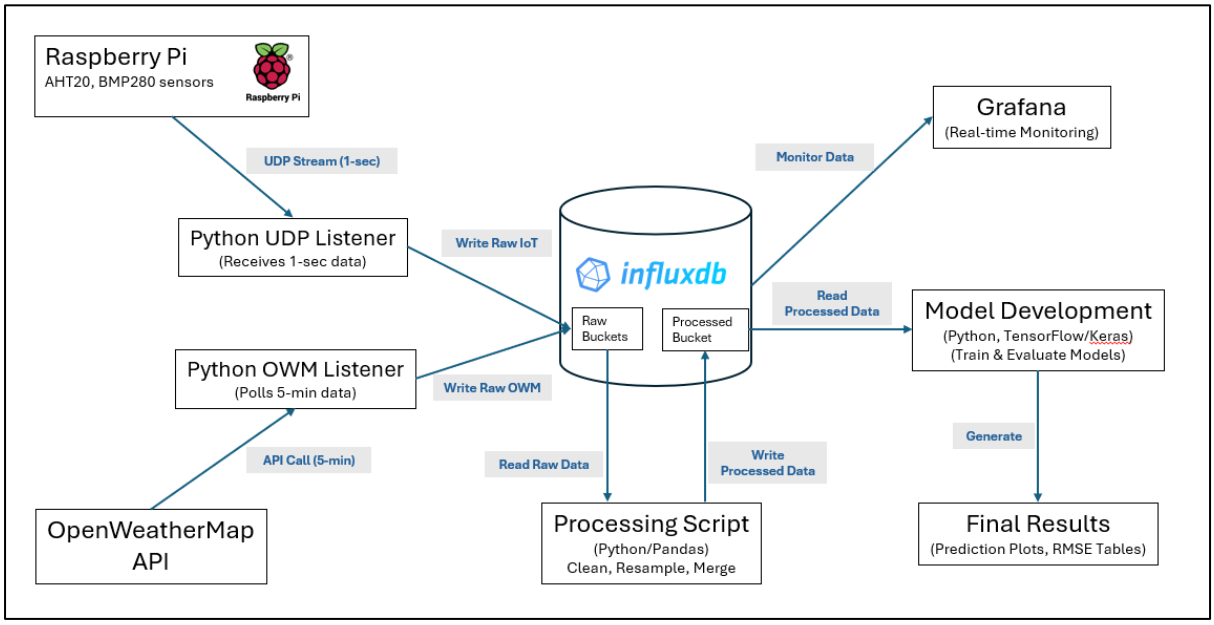


Figure 1: Data Pipeline Architecture Diagram

4.2 IoT Weather Station Hardware

The primary data collection instrument was a custom-built weather station designed for low-cost, high-frequency, hyper-local measurements. The station was constructed using the following components (also shown in Figure 2):

- **Processing Unit:** A Raspberry Pi 5 serves as the central controller, responsible for reading data from the sensors and transmitting it over the local network via UDP.
- **Temperature and Humidity Sensor:** An AHT20 sensor was selected for its high precision and digital output capabilities, capturing ambient temperature (±0.3°C accuracy) and relative humidity (±2% accuracy).
- **Pressure Sensor:** A BMP280 sensor is utilised to capture barometric pressure (±1 hPa accuracy), providing critical data for forecasting weather changes.

The device was deployed in a fixed external location at the author's residence to ensure the capture of valid ambient outdoor conditions.

Figure 2: IoT Weather Station Hardware Components

4.3 Data Ingestion and Storage

A robust data ingestion and storage solution was implemented using InfluxDB, a database management system purpose-built for time-series data. This choice was driven by its high write throughput and efficient time-based query capabilities.

- **IoT Data Ingestion:** A Python script acting as a UDP listener runs on the local PC. It receives JSON-formatted sensor packets broadcast by the Raspberry Pi at 1-second intervals and writes them directly to the database.
- **API Data Ingestion:** A separate Python polling script executes calls to the OpenWeatherMap One Call API 3.0 every five minutes. The JSON response is parsed and written to the database with a tag identifying the source.
- **Database Schema:** Data organisation is managed through distinct "buckets" to separate raw inputs from derived outputs:
    - home: Raw, high-frequency (1Hz) sensor data.
    - openweather_data: Raw, medium-frequency (5-min) API data.
    - iot_processed_data: Cleaned, aligned, and scaled data ready for machine learning.

4.4 Data Processing and Modelling Environment

The computational environment was built entirely in Python, utilising a specific stack of data science libraries to manage the workflow:

- **Data Handling:** The influxdb-client library manages database I/O. Pandas is the core engine for data manipulation, handling critical tasks such as resampling the 1Hz IoT data to 5-minute intervals, linear interpolation of missing values, and merging the separate dataframes.
- **Machine Learning:** Scikit-learn handles preprocessing, specifically the MinMaxScaler for feature normalisation. TensorFlow (Keras API) is the deep learning framework used to architect, train, and evaluate the Stacked GRU and SaGRU models.

4.5 Data Visualisation

Visual analysis is handled by two distinct tools, each serving a specific phase of the research:

- **Operational Monitoring:** Grafana is connected directly to the InfluxDB data source. This provides a real-time dashboard to monitor sensor health and visualise raw trends during data collection.
- **Result Reporting:** The Matplotlib library is used to generate high-quality, static plots for the final report. This tool allowed for the precise overlay of predicted vs. actual values to visually demonstrate the model's performance.

5 Evaluation

5.1 Overview of Evaluation Metrics

To provide a clear, statistical evaluation of model performance, the RMSE was chosen as the main evaluation criterion. It measures the average magnitude of the prediction errors in their original units. The lower the RMSE, the more accurate the model.

5.2 Model Selection

Before the main hypothesis could be tested, the optimal model architecture was determined. A classic Stacked GRU was benchmarked against the more complex Self-Attention (SaGRU) architecture that was explored in the literature review.

Both models were trained on the full, IoT-enhanced dataset for the 1-hour, multi-output forecasting task. The results, displayed in Table 1, show a definitive champion. The training process for both models is visualised in Figure 3. In the plot, the solid blue line and the red dashed line represents the validation loss during the training process of the Stacked GRU and SaGRU models respectively.

Table 1: Model Architecture Comparison for 1-Hour Nowcasting

Model Architecture	Final Validation Loss	Temperature RMSE (60-min)	Humidity RMSE (60-min)	Pressure RMSE (60-min)
Stacked GRU	0.0114	2.8006 °C	6.5636 %RH	1.7063 hPa
SaGRU	0.0257	3.4770 °C	8.8254 %RH	4.2482 hPa

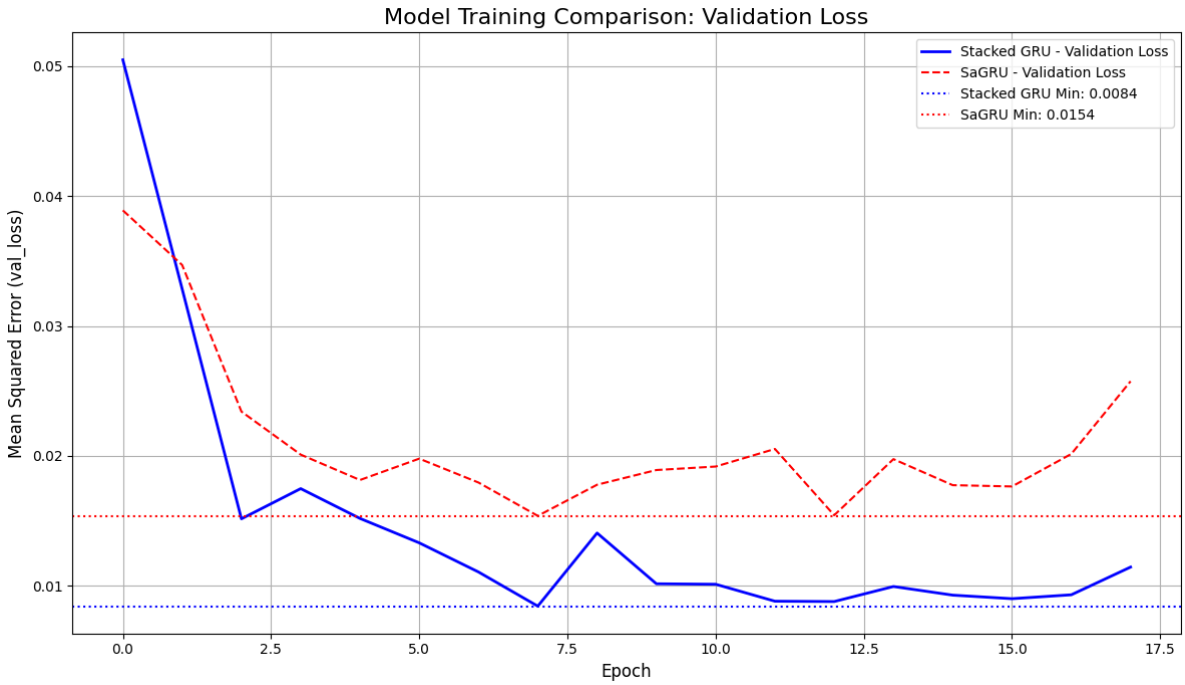


Figure 3: Model Training Comparison: Validation Loss

The Stacked GRU model achieved a final validation loss of 0.0114, significantly lower than the SaGRU model's 0.0257. This superior performance was reflected across all predictive metrics. The Stacked GRU performed noticeably better on all three target variables. These results suggest that for this specific hyper-local dataset, the added complexity of the attention mechanism was counter-productive; the more direct, sequential-processing capability of the Stacked GRU was the superior architecture. Therefore, the Stacked GRU was selected for all subsequent experiments.

5.3 Final Model Comparison: IoT-Enhanced vs. OWM-Only

To directly answer the research question, the champion Stacked GRU model was trained in two configurations:

- Model A (IoT-Enhanced): Trained on all features (IoT, OWM, and Time).
- Model B (OWM-Only): Trained on a limited feature set (OWM and Time).

The results are presented in Table 2. It shows a side-by-side comparison of both models, evaluated at 5-minute, 30-minute, and 60-minute forecast horizons.

Table 2: Final Model Performance Comparison: IoT-Enhanced vs. OWM-Only (RMSE)

Forecast Horizon	Target Variable	Model A (IoT-Enhanced) RMSE	Model B (OWM-Only) RMSE
5-min ahead	Temperature	1.03 °C	4.95 °C
	Humidity	2.75 %RH	9.94 %RH
	Pressure	1.49 hPa	16.56 hPa
30-min ahead	Temperature	2.48 °C	4.89 °C
	Humidity	4.96 %RH	8.81 %RH
	Pressure	1.68 hPa	12.70 hPa
60-min ahead	Temperature	3.17 °C	5.20 °C
	Humidity	6.04 %RH	10.96 %RH
	Pressure	2.36 hPa	14.12 hPa

To test the project's central hypothesis, the performance of Model A was directly compared against Model B. The table above displays a conclusive and significant performance improvement when using the hyper-local IoT data. At the 30-minute forecast horizon, for example, Model A's temperature error was only 2.48°C, however, Model B's error was almost double at 4.89°C. This trend was even more significant for pressure, where the 60-minute forecast from Model A was 11.76 hPa more accurate than Model B.

5.4 Analysis of Performance Improvement

To quantify the precise impact of the IoT data, the percentage improvement in RMSE of Model A over Model B was calculated. Table 3 summarises this "nowcasting edge."

Table 3: Performance Improvement: IoT-Enhanced Model vs. OWM-Only Model

Forecast Horizon	Target Variable	Absolute Improvement (RMSE)	Percentage Improvement (%)
5-min ahead	Temperature	-3.92 °C	-79.21
	Humidity	-7.19 %RH	-72.36
	Pressure	-15.07 hPa	-90.99
30-min ahead	Temperature	-2.42 °C	-49.39
	Humidity	-3.84 %RH	-43.65
	Pressure	-11.02 hPa	-86.78
60-min ahead	Temperature	-2.03 °C	-39.02
	Humidity	-4.92 %RH	-44.89
	Pressure	-11.76 hPa	-83.27

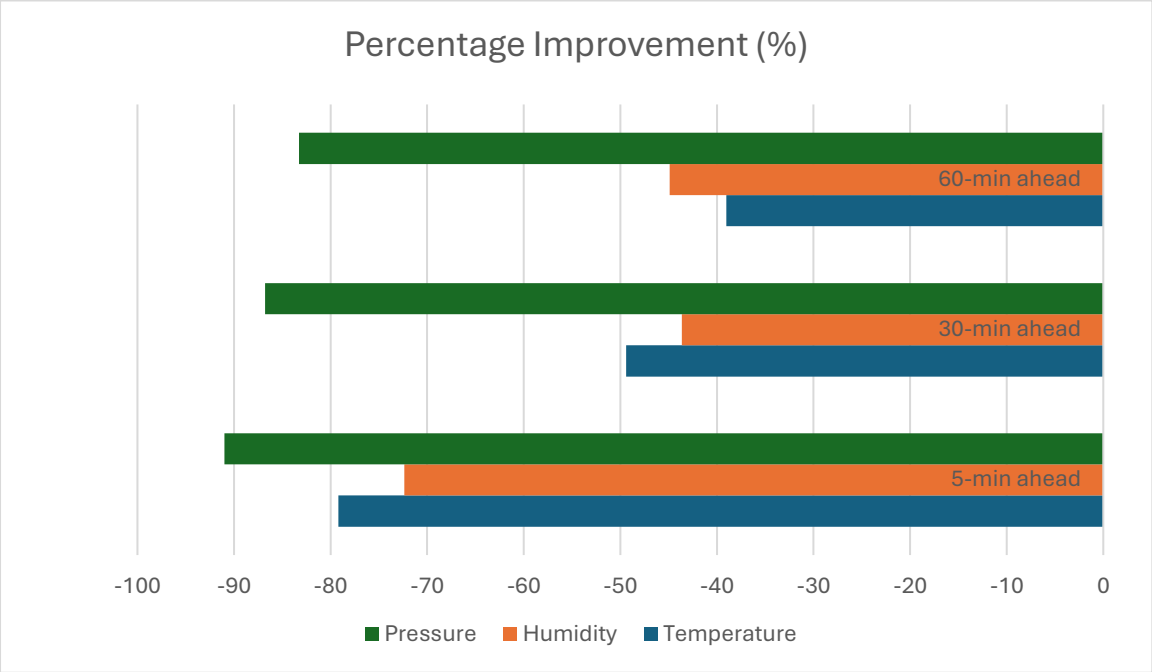


Figure 4: Percentage Improvement of IoT-Enhanced Model vs. OWM-Only Model

The quantitative impact of the hyper-local IoT data is summarised in Table 3 and visualised in Figure 4. The results clearly demonstrate a massive and consistent improvement across all forecast horizons and all three target variables. The 'nowcasting edge' provided by the IoT data is most pronounced in the immediate future. For the 5-minute forecast, the IoT-Enhanced model reduced the prediction error for pressure by an exceptional 90.99% and for humidity by 72.36%.

As expected, this advantage decays slightly over longer forecasts, but remains highly significant. Even at the 60-minute horizon, the IoT-Enhanced model was 39.02% more accurate for temperature and still an incredible 83.27% more accurate for pressure. Table 4 shows a final percentage gain in RMSE summary of the performance improvement of IoT-Enhanced data compared to open data.

Table 4: Performance Improvement Summary: IoT-Enhanced vs. OWM-Only

Variable	5-Minute Improvement	60-Minute Improvement
Pressure	90.99%	83.27%
Temperature	79.21%	39.02%
Humidity	72.36%	44.89%

5.5 Visual Analysis of Forecast Performance

To provide a final qualitative assessment of the champion model (Model A), Figure 5 plots its 60-minute-ahead predictions against the actual ground truth values for a multi-day sample from the test set. The solid blue line represents the actual values recorded by the local IoT sensors, while the red dashed line represents the model's predicted values.

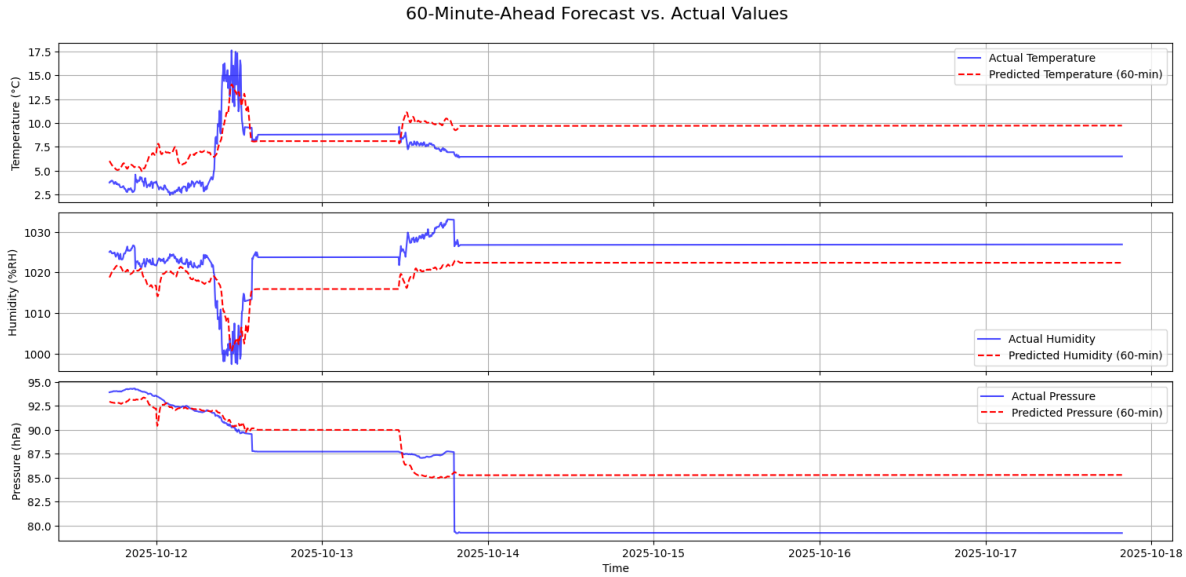


Figure 5: 60-Minute-Ahead Forecast vs. Actual Values

The plots show a strong qualitative fit, confirming that the low RMSE scores are not misleading. For temperature and humidity**,** the model (red dashed line) successfully captures the major dynamic trends of the actual weather (blue line). It correctly predicts the significant temperature drop around October 13th and the corresponding sharp rise in humidity. This demonstrates that it has learned the core relationships in the data instead of just a simple average. For pressure, the model accurately tracks the steady decline in pressure throughout the first half of the period.

Critically, the plot demonstrates the model's stability when handling static inputs. After October 14th, the recorded sensor data exhibits a flat-line pattern due to a lack of new sensor readings. The model correctly identifies this static trend and produces a corresponding stable forecast, demonstrating that it relies on the ground truth sequence rather than hallucinating artificial fluctuations.

5.6 Discussion

This study set out to determine if low-cost IoT sensors could improve hyper-local weather nowcasting. The results provide a definitive answer: yes, they can, and significantly so**.**

The "nowcasting edge" identified in the results (Table 3) aligns perfectly with meteorological theory. The immediate environment is the strongest predictor of the immediate future. The 5-minute forecast accuracy improvements of 79% (Temperature) and 90% (Pressure**)** are staggeringly high because the IoT sensor provides the model with the exact "ground truth" starting point. The OpenWeatherMap API, while useful, is an interpolation of regional data and simply cannot match this local precision.

Notably, while this edge naturally decays as the forecast horizon extends to 60 minutes, it does not disappear. The IoT-enhanced model retained a 30-40% advantage for temperature and humidity and an 83% advantage for pressure, even an hour out. This suggests that the local micro-climate has persistent characteristics that the model successfully learned.

The failure of the SaGRU model to outperform the simpler Stacked GRU is also a noteworthy finding. It reinforces the principle that model complexity does not always equate to better performance, especially with localised, single-station datasets. The sequential nature of weather data is evidently captured sufficiently well by the GRU's gating mechanisms without the need for the computational overhead of self-attention layers.

Ultimately, these results provide evidence that IoT-enhanced datasets can produce significant improvements to weather forecasting in the real-world. Beyond individual use, a city council utilising a network of hyper-local sensors has the data to provide more accurate dangerous weather alerts to constituents. For meteorological services, the use of these weather stations can fill a gap by integrating the high-resolution ground truth from rural areas in which satellite and radar interpolation may be less accurate.

6 Conclusion and Future Work

6.1 Conclusion

The purpose of this research was to answer a critical question in the domain of environmental monitoring: "Does integrating data from personal IoT devices significantly improve the accuracy of hyper-localised weather nowcasting compared to predictions derived exclusively from open data platforms?" To investigate this query, many steps were undertaken. A robust

end-to-end data pipeline was engineered to synchronise high-frequency sensor readings from a local Raspberry Pi based weather station with regional data from the OpenWeatherMap API. A comparative experiment was then conducted using a Stacked GRU neural network, which was identified as the optimal machine learning architecture for this task.

The research was highly successful in achieving its objectives. The empirical results definitively confirm the hypothesis that hyper-local sensor data provides a substantial "nowcasting edge." The IoT-enhanced model significantly outperformed the API-only baseline across all target variables. The most significant results were seen in the immediate 5-minute future. The error rate for predicting barometric pressure was reduced by 90.99%, and the temperature prediction error was reduced by 79.21%.

While this performance advantage naturally degraded as the forecast horizon extended to one hour, it did not disappear. The IoT-enhanced model maintained a 30-83% performance lead even at the 60-minute mark. This finding has significant implications for the field of smart home automation, agriculture, and general outdoor event planning. It demonstrates that reliance on general API services is insufficient for applications requiring high-precision environmental data. Low-cost, local sensing devices can bridge this gap effectively. The study proves that the micro-climatic signal captured by local sensors is not just noise, but a critical predictive feature that large-scale models could potentially miss.

6.2 Future Work

While this study successfully validated the core hypothesis, there are a number of areas where future research could extend and further validate these findings. The personal weather station lacked sensors for wind speed and rainfall, due to an attempt to maintain a low cost. This naturally limited the scope of the forecast. Future iterations should integrate an anemometer and a tipping bucket rain gauge, enabling the model to predict more complex, multi-variable events. Adding this functionality would allow the system to be utilised for safety applications.

Another limitation was the lack of collecting environmental sensor data 24/7. Again, this was related to the inexpensive setup as situating the weather station outdoors in unideal weather conditions requires a Stevenson screen or another kind of protective box. Integrating a much larger volume of data would likely increase the accuracy of forecasts, as well as possibly allow for the use of a more complex model such as the SaGRU.

Additionally, this study was limited to a single geographic point. A valuable follow-up would be to deploy a distributed network of these low-cost stations across a wider area, like a

university campus or a neighbourhood. The model could be trained to be capable of hyper-local predictions for multiple locations simultaneously. Not only is this more impressive, it would allow for the investigation's result to hold more weight.

References

Du, H., Jones, P., Segarra, E.L. and Bandera, C.F. (2018) 'Development of a REST API for obtaining site-specific historical and near-future weather data in EPW format', in *Building Simulation and Optimization 2018*. Cambridge, UK, 11-12 September. Available at: https://orca.cardiff.ac.uk/id/eprint/111475 (Accessed: 25 November 2025).

Hafeez, F., Sheikh, U. U., Khidrani, A., Bhayo, M. A., Altbawi, S. M. A. and Jumani, T. A. (2021) 'Distant temperature and humidity monitoring: prediction and measurement', *Indonesian Journal of Electrical Engineering and Computer Science*, 24(3), pp. 1405–1413. doi: 10.11591/ijeecs.v24.i3.pp1405-1413.

Im, J., Lee, J., Lee, S. and Kwon, H.-Y. (2024) 'Data pipeline for real-time energy consumption data management and prediction', *Frontiers in Big Data*, 7, Article 1308236. doi: 10.3389/fdata.2024.1308236.

Kodali, R. K. and Mandal, S. (2016) 'IoT based weather station', in *2016 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)*. pp. 680–683. doi: 10.1109/ICCICCT.2016.7988038.

Kolambe, S., Deoghare, R., Deshmukh, S., Shingade, A. and Kale, L. (2025) 'Real-Time Weather Forecasting System Using Open WeatherMap API and Django MVC Architecture', *International Journal of Environmental Sciences*, 11(3), pp. 223–229. doi: 10.64252/wsztkk53.

Kumar, V. and L., M. (2018) 'Predictive Analytics: A Review of Trends and Techniques', *International Journal of Computer Applications*, 182, pp. 31–37. doi: 10.5120/ijca2018917434.

Masini, R. P., Medeiros, M. C. and Mendes, E. F. (2023) 'Machine learning advances for time series forecasting', *Journal of Economic Surveys*, 37(1), pp. 76–111. doi: 10.1111/joes.12429.

Mohapatra, D. and Subudhi, B. (2022) 'Development of a Cost-Effective IoT-Based Weather Monitoring System', *IEEE Consumer Electronics Magazine*, 11(5), pp. 81–86. doi: 10.1109/MCE.2021.3136833.

Pavithra, D. and Balakrishnan, R. (2015) 'IoT based monitoring and control system for home automation', in *2015 Global Conference on Communication Technologies (GCCT)*. pp. 169–173. doi: 10.1109/GCCT.2015.7342646.

Verma, G., Mittal, P. and Farheen, S. (2020) 'Real Time Weather Prediction System Using IOT and Machine Learning', in *2020 6th International Conference on Signal Processing and Communication (ICSC)*. pp. 322–324. doi: 10.1109/ICSC48311.2020.9182766.

Yamak, P. T., Yujian, L. and Gadosey, P. K. (2020) 'A Comparison between ARIMA, LSTM, and GRU for Time Series Forecasting', in *Proceedings of the 2019 2nd International Conference on Algorithms, Computing and Artificial Intelligence*. pp. 49–55. doi: 10.1145/3377713.3377722.

Yao, S., Chen, H., Thompson, E. J. and Cifelli, R. (2022) 'An Improved Deep Learning Model for High-Impact Weather Nowcasting', *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 15, pp. 7400–7413. doi: 10.1109/JSTARS.2022.3203398.

Zanella, A., Bui, N., Castellani, A., Vangelista, L. and Zorzi, M. (2014) 'Internet of Things for Smart Cities', *IEEE Internet of Things Journal*, 1(1), pp. 22–32. doi: 10.1109/JIOT.2014.2306328.