International Journal of Computer Science Trends and Technology (IJCST) – Volume 12 Issue 1, Jan - Feb 2024

RESEARCH ARTICLE OPEN ACCESS

Atmosphere – A Weather Prediction & Monitoring System

Kaushik Kashyap [1], Rinku Moni Borah [2], Priyanku Rahang [3], Dr Bornali Gogoi [4], Prof. Nelson R Varte [5]

[1][2][3] MCA Student, Department of Computer Application, Assam Engineering College, Guwahati, Assam, India
[4][5] Associate Professor, Department of Computer Application, Assam Engineering College, Guwahati, Assam, India

ABSTRACT

This study looks at the collaborative use of Machine Learning (ML), the Internet of Things (IoT), and sophisticated neural networks for weather monitoring and prediction. The goal is to create a flexible system that integrates IoT sensors, machine learning algorithms, and web development to improve real-time weather forecasting. For time series forecasting, the project utilizes a variety of neural network designs, including Convolutional and Recurrent Neural Networks, with a concentration on the Long Short-Term Memory (LSTM) model. The ESP8266 NodeMCU is used for real-time data collecting in the IoT implementation, while the ML model goes through painstaking data preparation, feature extraction, and time series forecasting. The paper finishes with the successful integration of the machine learning model into the IoT system for both real-time and anticipated weather data.

***Keywords*:** ESP8266 NodeMCU, DHT22 Sensor, CNN, LSTM, Mean Squared Error (MSE).

I. INTRODUCTION

A major development with possibility for improving weather monitoring and prediction systems exists at the intersection of Machine Learning (ML), the Internet of Things (IoT), and sophisticated neural networks. Traditional systems frequently fall short of delivering real-time and localized data, necessitating the development of novel alternatives. This study aims to combine IoT sensor technology, ML algorithms, and web development to produce a dynamic system for accurate weather forecasting. Motivated by the limits of existing systems, we are working to create a flexible platform capable of not only collecting and displaying environmental data, but also utilizing predictive analytics. The use of ML methods such as Convolutional and Recurrent Neural Networks is investigated, with the ESP8266 NodeMCU acting as the IoT backbone.

II. LITERATURE REVIEW

A. Supervised Learning

In the realm of supervised learning, the training of a model relies on paired input data and corresponding output labels within a labelled dataset. For this architecture, the suitability of supervised learning hinges on the availability of a dataset containing input sequences alongside their respective target values. In this context, the Mean Squared Error (MSE) loss function is deemed appropriate, calculating the average squared difference between predicted and observed values.

B. Unsupervised Learning

Conversely, unsupervised learning operates on datasets devoid of explicit output labels, focusing on identifying structures, connections, or patterns within the data. While the architecture primarily caters to supervised learning tasks, components like the bidirectional processing of LSTM layers and convolutional layers can find utility in unsupervised feature learning or clustering scenarios.

C. Neural Networks

Neural networks, inspired by biological neural networks, are a class of machine learning models. Comprising interconnected nodes or neurons organized in layers, these networks excel in learning and modelling intricate relationships. Tasks such as speech recognition, image recognition, and prediction align well with their capabilities.

D. Artificial Neural Networks (ANNs)

Artificial Neural Networks (ANNs) emulate the structure of the human brain, featuring layers of interconnected neurons, including input, hidden, and output layers. Training ANNs involves optimizing the network's performance by adjusting the weights and biases of the neurons.

E. Convolutional Neural Networks (CNNs)

Specialized for processing grid-like data like images, Convolutional Neural Networks (CNNs) prove highly effective in tasks such as image classification and recognition. Incorporating convolutional layers for feature extraction and pooling layers for spatial down sampling, CNNs excel in capturing spatial relationships in images.

F. Recurrent Neural Networks (RNNs)

Tailored for sequential data like time series or text, Recurrent Neural Networks (RNNs) leverage feedback connections to retain and utilize information from previous steps. This makes them ideal for tasks involving context and temporal dependencies, including language modelling, speech recognition, and image captioning.

G. Long Short-Term Memory (LSTM)

The Long Short-Term Memory (LSTM) architecture, within the broader context of Recurrent Neural Networks, addresses the vanishing gradient problem encountered in regular RNNs trained on extended sequences. Through gating mechanisms and memory cells, LSTMs selectively remember or forget information, making them well-suited for applications requiring sequential patterns, time series prediction, and natural language processing. The design's input, output, and forget gates regulate information flow through the cell over time.

H. ESP8266 NodeMCU

The ESP8266 NodeMCU represents an iconic leap in IoT solution development. It is a versatile and cost-effective development board that combines the capabilities of the ESP8266 Wi-Fi module with the ease of use of NodeMCU software. This combination results in a strong platform that allows researchers and developers to easily build wireless communication. The ESP8266 NodeMCU acts as a cornerstone in applications ranging from home automation to sensor networks, offering a basis for efficient and networked devices. Its small size conceals its powerful capabilities, making it a key component in the execution of revolutionary Internet of Things concepts.

I. DHT22 Sensor

The DHT22 sensor is a high-precision temperature and humidity measurement tool. The DHT22, known for its precision and durability, uses digital signal output to offer real-time data with little mistake. Its use in research initiatives improves their capacities by providing precise and timely environmental data. The DHT22 sensor becomes an important tool whether used in temperature control systems, agricultural research, or smart home applications, recording and conveying crucial information that enables educated decision-making and automation.

III. METHODOLOGY

A. Proposed Method

The study takes a comprehensive approach to constructing a real-time weather monitoring and forecast system by integrating sensors and machine learning algorithms. Using sensors, the system actively collects real-time meteorological data, which is then processed and evaluated using deep learning techniques, namely the Long Short-Term Memory (LSTM) model. This technique allows the system to provide accurate seven-day temperature forecasts by orchestrating an architecture that smoothly blends data processing and machine learning for effective real-time weather monitoring and prediction.

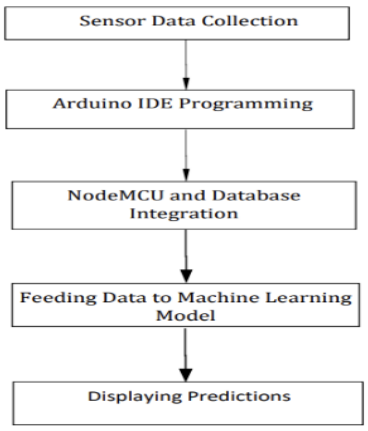

Sensor Data Collection
Arduino IDE Programming
NodeMCU and Database Integration
Feeding Data to Machine Learning Model
Displaying Predictions

Figure 2.1: Block Diagram of AtmoSphere

B. Data Collection

Initiating with historical weather data acquisition, the recognition system employs a dataset sourced from Kaggle. Sensors capture environmental data at regular intervals, generating a continuous stream of measurements. The collected data is merged with the historical dataset for comprehensive analysis.

C. Pre-processing

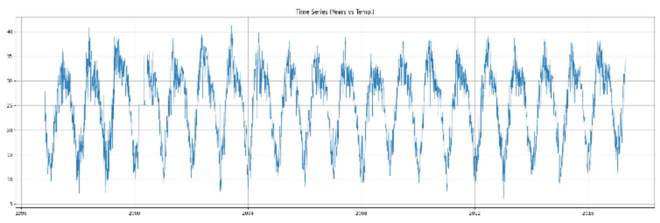
The data pre-processing stage entails arranging gathered weather data into certain forms and separating it into training, testing, and validation datasets. Standardization is used to calculate the average and standard deviation of these datasets. Before being sorted into a specific format for analysis and model training, the data travels through many forms to test prediction performance.

D. Feature Extraction

Feature extraction is critical in the development of a comprehensive weather monitoring and prediction system because it identifies and selects the most relevant factors that contribute to weather changes. This stage guarantees that the model is trained on the dataset's most relevant and informative features.

E. Time Series Forecasting

Matplotlib in Python is employed for a comprehensive visualization of the dataset. The time series plot illustrates temperature evolution over the years, presenting a clear depiction of temporal trends. This visual representation complements subsequent discussions on the



neural network architecture.

Figure 2.2.1: Time Series Forecasting Graph

F. Model Architecture

A sophisticated neural network architecture is designed for sequential data processing, combining local pattern recognition and capturing long-term interdependence. This hybrid model incorporates components from both a recurrent neural network (RNN) and a convolutional neural network (CNN).

1. Convolutional Layers

Initiating with two Conv1D layers, local patterns in the input sequence are identified. The first layer employs 256 filters, followed by 128 filters in the second layer. Small kernels (size 2) and Rectified Linear Unit (ReLU) activation function capture complex patterns.

2. Pooling Layer

A MaxPooling1D layer with a pool size of 2 down samples spatial dimensions from the convolutional layers, reducing computational load without compromising vital information.

3. Flatten and Repeat Vector

After convolutional and pooling layers, a Flatten layer converts the output into a 1D vector. A Repeat Vector layer replicates this vector 30 times, a common strategy in sequence-to-sequence tasks requiring output for each element in the input sequence.

4. LSTM Layers

Three 100-unit Long Short-Term Memory (LSTM) layers capture long-term dependencies in sequential data, training hierarchical representations. Dropout layers, with a rate of 0.2, mitigate overfitting.

5. Bidirectional LSTM

A Bidirectional wrapper applied to an LSTM layer with 128 units enables processing the input sequence in both forward and backward directions, enhancing contextual capture.

6. Dense Layers

Post-LSTM layers, a Dense layer with 100 units and ReLU activation provides a high-level representation. The final Dense layer with 1 unit serves as the output layer for regression tasks, predicting a single value for continuous variables.

7. Model Compilation

Compiled with Mean Squared Error (MSE) loss function and Adam optimizer, the model dynamically adjusts learning rates during training for effective gradient descent.

8. Callbacks

Enhancing training efficiency, an Early Stopping callback monitors training loss and halts training after a set number of epochs (patience=7), minimizing overfitting risks.

This methodology lays the groundwork for an advanced weather monitoring and prediction system, integrating data collection, pre-processing, feature extraction, and a hybrid neural network model for robust forecasting capabilities. The architecture seamlessly combines information processing and machine learning, forming an effective framework for real-time weather monitoring and prediction.

IV. IMPLEMENTATION

A. ESP8266 NodeMCU Workflow:

1. **Sensor Data Acquisition:** The DHT22 sensor connected to the NodeMCU module collects temperature and humidity data from the environment.

2. **Arduino IDE Programming:** Arduino IDE is used to write the code that instructs the NodeMCU on how to interact with the DHT22 sensor. This code includes instructions for reading sensor data and formatting it for transmission.

3. **NodeMCU and XAMPP Integration:** The NodeMCU module is programmed to communicate with the XAMPP server. It sends the collected sensor data to the server for storage.

4. **Data Storage on XAMPP Server:** The XAMPP server receives the data from the NodeMCU and stores it, creating a log of temperature and humidity readings over time. Data stored on the XAMPP server can be accessed and monitored remotely from anywhere with internet access.

B. Building the Machine Learning Model

It involves a series of steps, including data preparation, model building, training, and evaluation. The overall process can be summarized as follows:

i. Data Preparation:

1. **Loading and Cleaning Data:** The necessary libraries are imported, and the dataset containing historical temperature data is loaded from a specified location. Missing values are handled appropriately, and the data is pre-processed to ensure its suitability for modeling.

2. **Exploratory Data Analysis:** The data is analysed to gain insights into the

3. distribution of temperature values, the frequency of different weather conditions, and the relationship between temperature and other factors such as time of year and wind direction.

4. **Feature Engineering:** New features, such as year and month, are extracted from the date information to better capture temporal patterns in the data.

ii. Model Building and Training:

1. **Time Series Forecasting:** The temperature feature is extracted from the dataset, along with the corresponding date information as the index. The data is resampled to ensure consistent time intervals.

2. **Scaling Data:** To address the issue of outliers and ensure numerical stability, the temperature data is scaled using MinMaxScaler. This process transforms the data into a range between -1 and 1.

3. **Creating Training and Testing Sets:** The scaled data is split into training and testing sets. The training set is used to train the model, while the testing set is used to evaluate its performance.

4. **Model Definition**: A Long Short-Term Memory (LSTM) model is defined. LSTM is a type of recurrent neural network (RNN) well-suited for time series forecasting tasks. The model architecture consists of multiple LSTM layers, along with other layers like convolutional and dense layers, to capture complex temporal patterns and relationships in the data.

5. **Model Compilation:** The LSTM model is compiled using the mean squared error (MSE) loss function and the Adam optimizer.

6. **Model Training:** The model is trained on the training set for a specified number of epochs. Early stopping is implemented to prevent overfitting and improve the model's generalization.

iii. Model Evaluation and Prediction:

1. **Model Evaluation:** The trained model is evaluated on the testing set. The mean squared error (MSE) and R-squared (r2) metrics are used to assess the model's performance.

2. **Making Predictions:** The model is used to make predictions for future temperature values. The predictions are then transformed back to the original scale using the inverse transform of the MinMaxScaler.

3. **Visualizing Results:** The predicted temperature values are compared to the actual temperature values from the testing set to visualize the model's performance.

By following these steps, the model is evaluated and found to perform well on the testing set, demonstrating its ability to generalize and make accurate predictions.

b. Integration with IoT System

This model is seamlessly incorporated into the pre-specified IoT system through the modification of NodeMCU firmware. It transmits historical temperature data to the LSTM model, through which predictions are generated. Subsequently, these predictions are retrieved and securely stored in the XAMPP server.

c. Web Interface Design

A visually appealing interface is designed for the display of both real-time and predicted data using **HTML, CSS, and JavaScript**.

1. **Real-Time and Predicted Data Display**: JavaScript is implemented to fetch and showcase real-time sensor data, along with the presentation of predicted minimum and maximum temperatures.

2. **Testing:** The system is tested to ensure the accuracy of predicted values and effective handling of any errors or edge cases.

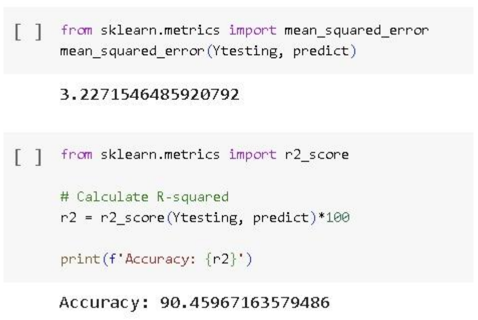
V. RESULT

A. Model Accuracy

Model accuracy is a pivotal metric, gauging the precision of a predictive model by comparing its output to actual values. A high accuracy signifies close alignment between model predictions and real-world outcomes. In the case of the AtmoSphere LSTM model, an impressive accuracy rate of 90.459% is achieved.

B. Model Evaluation and Prediction

The evaluation employs Mean Squared Error (MSE), a key performance metric in machine learning that quantifies the average squared difference between predicted and actual values in regression problems. Computed by summing squared residuals and dividing by the number of observations, MSE offers insight into the model's accuracy. A lower MSE indicates superior performance with minimal prediction errors. The LSTM model exhibits a commendable 3.227 Mean Squared Error value.



```
[ ]  from sklearn.metrics import mean_squared_error
     mean_squared_error(Ytesting, predict)

     3.2271546485920792

[ ]  from sklearn.metrics import r2_score

     # Calculate R-squared
     r2 = r2_score(Ytesting, predict)*100

     print(f'Accuracy: {r2}')

     Accuracy: 90.45967163579486
```

Figure 7.2.1: MSE and Accuracy of the model

C. Visualization

Critical to model evaluation, a graphical representation is generated to visually compare actual temperature values (Y_testing) in blue with corresponding predicted temperature values (predict) in red. This graphical illustration facilitates a comprehensive understanding of the model's performance, allowing immediate visual inspection of alignment between predicted and actual data points. The visualization reveals patterns, trends, and potential discrepancies, contributing to an intuitive and insightful interpretation of the model's predictive capabilities.

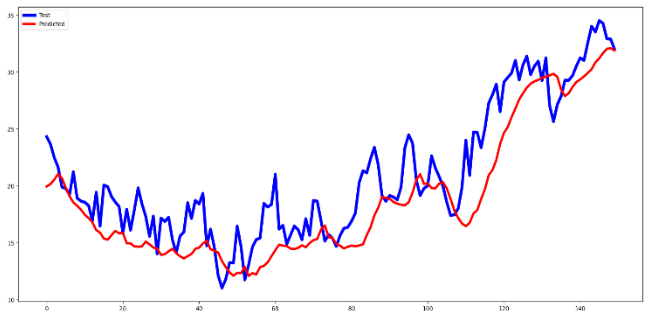


Figure 5.1: Forecast Results of the LSTM Model

D. Predictions

The model provides temperature and humidity forecasts for the next 7 days. Analysing historical temperature data, the model discerns trends and patterns, enabling accurate and tailored predictions for the immediate future. This data-driven forecasting mechanism ensures adaptation to dataset nuances, enhancing the model's precision in forecasting temperatures.

The LSTM model exhibits outstanding accuracy (90.459%) and minimal Mean Squared Error (3.227). Visualizations enhance interpretability, showcasing the model's prowess in forecasting, making it a robust tool for real-time weather predictions.

VI. CONCLUSION

This study explores a comprehensive approach to weather monitoring and prediction by combining Machine Learning (ML), the Internet of Things (IoT), and sophisticated neural networks. Our adaptable solution enables real-time weather forecasting by combining IoT sensors, ML algorithms, and web development. Our design shines by using Convolutional and Recurrent Neural Networks, with a focus on the Long Short-Term Memory (LSTM) model. The ESP8266 NodeMCU collects data in real time, while careful data preparation, feature extraction, and time series forecasting improve ML model performance. The successful integration of ML into the IoT system creates a solid framework for both real-time and forecast meteorological data. The suggested approach represents a big step forward in the development of precise and adaptable weather forecasting, promoting improvements in environmental intelligence.

REFERENCE

[1] Huang, Zi-Qi; Chen, Ying-Chih; and Wen, Chih-Yu, “Real-Time Weather Monitoring and Prediction Using City Buses and Machine Learning. Sensors”, (2020).

[2] Patkar, Uday. “Weather Prediction Using Machine Learning”, (2022).

[3] Khoa Lai, “Time Series Analysis and Weather Forecast in Python”, (2020).

[4] Random Nerd Tutorial. ESP8266 DHT11/DHT22 Temperature and Humidity Web Server with Arduino IDE

[5] Taron Foxworth, “Getting Started with the ESP8266 and DHT22 Sensor”, (2017).

[6] Mark Holmstrom; Dylan Liu; and Christopher Vo, “Machine Learning Applied to Weather Forecasting”, (2016).

[7] Gaurav Verma; Pranjul Mittal; and Shaista Farheen, “Real Time Weather Prediction System Using IOT and Machine Learning”, (2020).

[8] Anubha Parashar, “IoT Based Automated Weather Report Generation and Prediction Using Machine Learning”, (2019).

[9] C.K.Gomathy; and V.Geetha, “Weather Forecasting Application Using Python” (2022).