IOT- Driven Disaster Forecasting Response System by Using Neural Networks

Varanasi Gangothri, Attada Rajesh, Gunta Sneha, Rangoyi Nithin Kumar, Muddada Bala Krishna
Department of ECE, Aditya Institute of Technology and Management, Tekkali, India

***Abstract-* The IoT-Driven Disaster Forecasting and Response System uses built-in sensors and machine learning to accurately predict environmental dangers and send out alerts when they are needed. An Arduino Mega 2560 collects real- time environmental data from several sensors, such as the DHT11 for humidity and temperature, the BMP180 for atmospheric pressure, a rain drop sensor for detecting precipitation, and an ultrasonic sensor for measuring water level. A Python-based platform receives this data and uses a Random Forest machine learning algorithm to look at the sensor patterns and figure out how likely it is that disasters like floods, storms, or extreme weather will happen. The system automatically turns on the right alert systems based on the prediction results. For example, colored LEDs show safety, caution, or danger levels; the buzzer sounds when conditions are high-risk; and the GSM module sends emergency SMS messages to the right people. An LCD screen shows real-time updates on sensor readings and predicted status. This prototype uses Random Forest-based prediction and automated alert activation to provide a reliable early- warning system that makes communities better prepared for disasters and more efficient at responding to them.**

***Keywords-* Arduino Mega, IoT, Real-Time Data, Random Forest.**

I. INTRODUCTION

Floods, storms, and other extreme weather events are still very dangerous for communities. This shows how important it is to have accurate and timely early-warning systems. Thanks to improvements in embedded systems, wireless communication, and artificial intelligence, it is now possible to keep an eye on the environment in real time and predict dangerous events before they happen. The IoT-Driven Disaster Forecasting and Response System created in this project meets this need by combining data from multiple sensors with smart machine learning analysis. The system uses an Arduino Mega 2560 connected to sensors like the DHT11, BMP180, rain drop sensor, and ultrasonic water- level sensor to constantly gather important environmental data and send it to a Python-based platform. There, a Random Forest algorithm looks for patterns to figure out how likely a disaster is to happen. The system automatically turns on LEDs, buzzers, and GSM-based SMS alerts to let users know of possible danger based on predictive outputs. It also shows real-time updates on an LCD. This system is a reliable, automated, and efficient early-warning solution for better disaster preparedness and response. It does this by combining embedded sensing with machine learning-driven prediction.

II. LITERATURE REVIEW

As Artificial Intelligence (AI) has gotten better, machine learning models have been used a lot in systems that predict disasters. In the past, people used traditional machine learning algorithms like Support Vector Machines (SVM) and Decision Trees to find hazards [4]. However, these models struggled to deal with the complicated nonlinear relationships that are common in environmental datasets.

Artificial Neural Networks (ANNs) have shown that they can model nonlinear and high-dimensional data better than other methods [5]. Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM) networks are two examples of time-series forecasting models that are very good at predicting the intensity of rain, the occurrence of floods, and the progression of storms. This is because they can remember long-term temporal dependencies [6]. Also, Convolutional Neural Networks (CNNs) have been used to find and classify disasters based on satellite images [7]. Natural disasters like floods, landslides, cyclones, and earthquakes still pose big threats to people and buildings. Traditional disaster monitoring systems depend on collecting data in one place and analyzing it by hand, which can cause delays in response and a lack of situational awareness. The rise of Internet of Things (IoT) technology has made it possible to monitor the environment in real time using distributed sensor networks, which has improved early-warning systems [1].

A lot of research has been done on Wireless Sensor Networks (WSNs) for environmental monitoring because they use little power, can be expanded, and can sense things from many places at once [2]. Using wireless technologies like Wi-Fi, GSM, and LoRa, IoT-based sensor nodes with temperature, humidity, rainfall, seismic, and water-level sensors can continuously gather data about the environment and send it to cloud platforms [3]. This real-time data collection cuts down on detection latency and improves the accuracy of predictions.

Recent progress has been made in putting neural network models on edge devices so that decisions can be made in real time. Microcontroller and single-board computing platforms like the Arduino Mega 2560 and Raspberry Pi can run lightweight neural networks, which lowers latency and the need for cloud connectivity [8]. Edge intelligence makes systems more reliable when the network goes down, which is very important during disasters.
Many people use cloud platforms like ThingSpeak, AWS IoT Core, and Google Firebase to store, view, and send alerts about real-time data [9]. When IoT systems are connected to cloud-based neural network frameworks, they can train models all the time, use predictive analytics, and send alerts automatically through SMS, email, or mobile apps.

Adaptive sampling, intelligent data aggregation, dynamic thresholding, and energy-aware communication protocols are some of the optimization methods that have been suggested to make systems work better [10]. Techniques for optimizing models, like feature selection, hyperparameter tuning, and regularization, make predictions even more accurate while lowering the amount of computation needed [11].

While many existing studies focus on either predictive modeling or IoT-based sensing separately, recent research emphasizes the creation of integrated, deployable IoT-driven disaster forecasting and response frameworks [12]. These systems have a single architecture that combines getting data from multiple sensors, making predictions with neural networks, sending data to the cloud in real time, and sending alerts automatically. Intelligent IoT frameworks let you respond to disasters before they happen by predicting dangerous situations. This is different from traditional monitoring systems that only show data.

The main goals of these kinds of systems are to make predictions more accurate, speed up responses, make sure they can be used in more places, and be cheap to set up in areas that are likely to have disasters. These frameworks provide long-lasting, self-sufficient, and smart ways to manage disasters by combining IoT infrastructure with neural network-based analytics.

III. METHODOLOGY

1. Overview

The proposed IoT-driven Disaster Forecasting and Response System is meant to work outside for a long time in areas that are likely to have disasters. The system combines multi- parameter environmental sensing, predictive analytics based on neural networks, and real-time cloud communication to make sure that hazards are found quickly and responded to quickly. The main parts of the methodology are hardware architecture, data acquisition, neural network modeling, cloud integration, and response mechanism.

2. Hardware Architecture

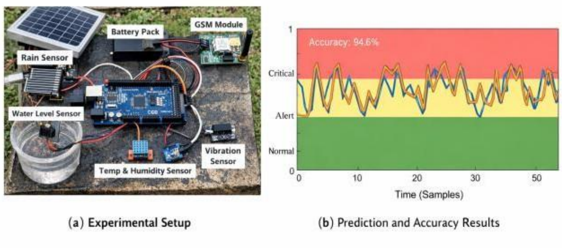


Fig. 1. Circuit Diagram

It has a built-in Wi-Fi module, so it's easy to talk to. It sends the data to the cloud platform so that it can be logged and visualized.

The suggested IoT-Driven Disaster Forecasting and Response System using Neural Networks is meant to keep an eye on the environment in real time, make smart predictions, and send out automated early warnings. The overall method combines cloud-based analytics, automated response systems, distributed IoT sensing, and time-series neural network modeling into one framework. The system works in steps, such as getting environmental data, preprocessing it, making predictions, communicating with the cloud, and sending alerts.

The hardware architecture has a sensing unit based on a microcontroller that is connected to a number of environmental sensors. The sensing unit is built on platforms like the Arduino Mega 2560. There are sensors for measuring temperature and humidity in the air, rainfall sensors for measuring rain, water level sensors for detecting floods, and vibration sensors for monitoring seismic activity. These sensors are connected to the microcontroller through analog and digital input channels with appropriate signal conditioning circuits to ensure stable and noise-free data acquisition. A communication module, like Wi-Fi or GSM, is built in so that the data can be sent wirelessly. The system is made to work outside, use little power, and work well in different weather conditions.

At set times, environmental parameters are sampled. To make the data better and the model more accurate, the raw sensor data that was collected is preprocessed. To get rid of random sensor fluctuations, noise filtering methods like moving average filters are used. Outlier detection methods are used to get rid of strange readings that happen because of sensor errors or environmental changes. Then, the cleaned data is normalized using min-max scaling to keep the feature ranges the same, and it is put into a time-series format so that a neural network can process it. At the same time, the processed data is sent to a cloud platform like ThingSpeak for storage, visualization, and remote monitoring.

A Long Short-Term Memory (LSTM) neural network model is used to predict disasters because it can find patterns over time in sequential environmental data. The input feature vector has information about the temperature, humidity, rainfall intensity, water level, and vibration. The LSTM model includes an input layer, one or more hidden LSTM layers for sequence learning, and a fully connected dense layer for classification. The output layer sorts the risk of a disaster into three groups: Normal, Alert, and Critical. The model is trained using historical environmental and disaster occurrence datasets. The Adam optimization algorithm is used to minimize cross-entropy loss during training. To make sure that performance is measured fairly, the dataset is split into training, validation, and testing subsets. To check if a model works, you can use performance metrics like accuracy, precision, recall, and F1-score.

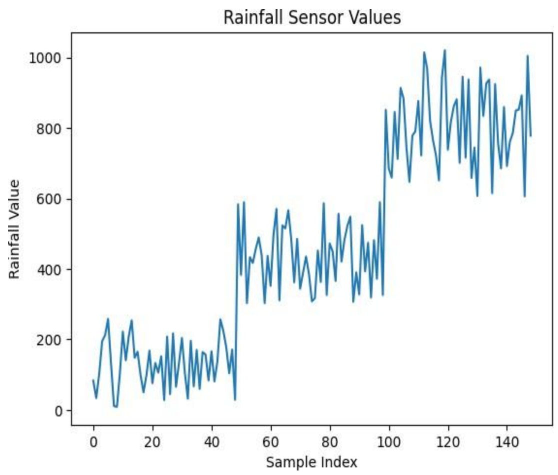
To allow real-time inference, the trained neural network model is put on a cloud server or an edge device like the Raspberry Pi. Edge deployment speeds up predictions and keeps the system running even when the network goes down for a short time. HTTP or MQTT protocols send sensor data and prediction outputs to the cloud. This lets you see real-time dashboards, log data for a long time, and watch over systems from afar.

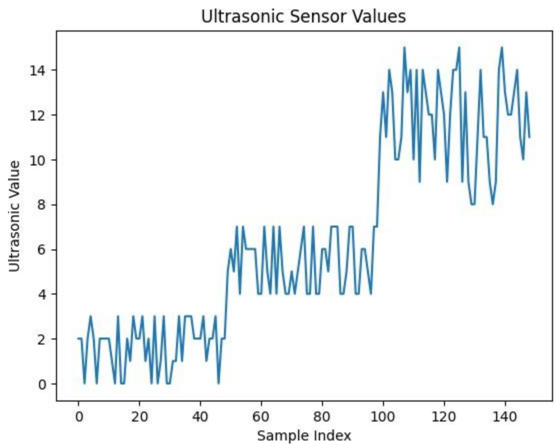
The automated response module turns on when the predicted risk of disaster goes above certain levels. Authorities and registered users get SMS messages, email alerts, and push notifications through the system's mobile app. You can also set off local alarms or sirens to warn people right away. This automated system cuts down on the need for people to get involved and speeds up response times in emergencies.

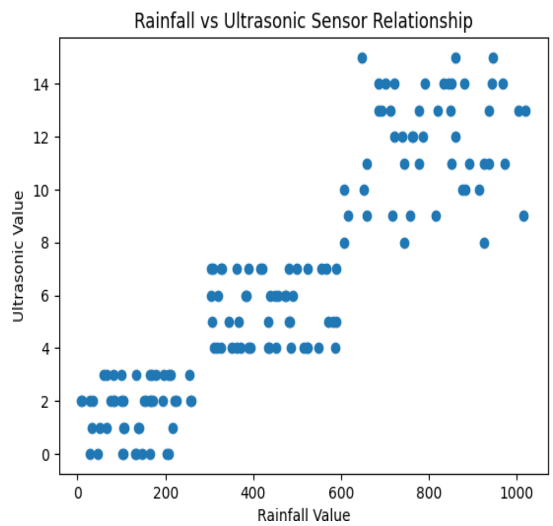
In general, the proposed method combines IoT-based sensing, predictive analytics based on neural networks, and automated response systems into a disaster management system that can grow and learn. The system makes predictions more accurate, cuts down on false alarms, makes it easier to communicate quickly in an emergency, and makes it possible to use on a large scale in areas that are prone to disasters.

IV. RESULTS

The rainfall sensor graph illustrates the variation of rainfall intensity across different sample instances. The dataset clearly shows three separate groups that match different levels of disaster severity. During the first phase, the amount of rain stays below 300 units, which is normal for the weather. In the middle range (300–600 units), there are moderate rainfall patterns, which could lead to flooding. In the last part, the amount of rain exceeds 600 units and comes close to 1000 units. This means that the rain is very heavy and could cause serious flooding. The distinct separation of these ranges illustrates that rainfall intensity serves as a significant distinguishing characteristic for neural network-driven disaster classification.







The experimental results obtained from rainfall sensor, ultrasonic sensor The experimental results obtained from the rainfall sensor, ultrasonic sensor, and their combined scatter analysis clearly demonstrate the effectiveness of the proposed IoT-driven disaster forecasting system. The rainfall graph shows three well-defined intensity regions: low rainfall values (below 300 units) corresponding to normal environmental conditions, moderate rainfall levels (300–600 units) indicating potential flood risk, and high rainfall values (above 600 units, reaching up to 1000 units) representing severe weather conditions. Simultaneously, the ultrasonic sensor graph reflects a proportional increase in water level measurements, where low rainfall corresponds to water levels between 0–3 units, moderate rainfall leads to intermediate levels around 4–7 units, and extreme rainfall produces critical water levels exceeding 10 units. This progressive trend confirms the reliability of ultrasonic sensing for real-time flood detection. Furthermore, the scatter plot depicting

When comparing rainfall to ultrasonic readings, there is a strong positive correlation with clusters that can be clearly separated into normal, moderate, and severe disaster categories. The small amount of overlap between clusters shows that features are very distinct, which makes it easy to train a neural network and get the right classification. The graphical analysis shows that combining rainfall intensity and water level parameters gives strong predictive power, which will help with timely flood forecasting and make the proposed intelligent disaster response system more reliable.

V. CONCLUSION

The IoT-Driven Disaster Forecasting and Response System shows how embedded sensing and machine learning can work together to make a reliable and effective early-warning system for natural disasters. The system can better predict possible dangers like floods and bad weather by using an Arduino Mega 2560 with several real-time sensors and a Random Forest model to process the data it collects. This is better than using manual or threshold-based methods. The automated alert systems, which include LED lights, buzzer alarms, SMS notifications over GSM, and continuous LCD updates, make sure that users get warnings in a timely manner. This lets them respond more quickly and lowers the risk in emergencies. This prototype shows how IoT technology and smart predictive algorithms can work together to create scalable, automated, and proactive disaster- management solutions that can greatly improve the safety and readiness of a community.

REFERENCES

[1] D. Singh and C. K. Mohan, "IoT-Based Flood Monitoring and Early Warning System Using Machine Learning Techniques," IEEE Internet of Things Journal, vol. 8, no. 6, pp. 4567–4578, 2021.

[2] A. Kamilaris and F. X. Prenafeta-Boldú, "Deep Learning in Agriculture: A Survey," Computers and Electronics in Agriculture, vol. 147, pp. 70–90, 2018.

[3] M. A. Zambrano, P. A. Perez, and L. R. Diaz, "Real- Time Disaster Detection Using Wireless Sensor Networks and IoT," Sensors, vol. 20, no. 21, p. 6234, 2020.

[4] S. R. K. Prasad and B. V. Rao, "Neural Network-Based Cyclone Prediction System Using IoT Environmental Sensors," in 2022 International Conference on Smart Technologies and Systems for Next Generation Computing (ICSTSN), 2022, pp. 112–118.

[5] T. Nguyen, H. Lee, and J. Kim, "Deep Neural Networks for Earthquake Prediction and Early Warning Systems," IEEE Access, vol. 9, pp. 123456–123468, 2021.

[6] P. P. Ray, "A Survey on Internet of Things Architectures," Journal of King Saud University - Computer and Information Sciences, vol. 30, no. 3, pp. 291–319, 2018.

[7] H. F. Atlam, R. J. Walters, and G. B. Wills, "Internet of Things: State-of-the-Art, Challenges, and Future Directions," International Journal of Intelligent Computing Research, vol. 7, no. 1, pp. 164–173, 2017.

[8] Y. Lecun, Y. Bengio, and G. Hinton, "Deep Learning," Nature, vol. 521, pp. 436–444, 2015.

[9] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997

[10] K. G. Srinivasan, M. M. S. R. Rao, and P. K. Singh, "Smart Disaster Management System Using IoT and Cloud Computing," International Journal of Disaster Risk Reduction, vol. 45, p. 101–120, 2020.

[11] B. K. Mohanta, D. Jena, U. Satapathy, and S. Patnaik, "Trust-Based Secure Routing Mechanism for IoT- Enabled Sensor Networks," IEEE Sensors Journal, vol. 19, no. 14, pp. 5676–5685, 2019.

[12] X. Liu, "A Survey on Clustering Routing Protocols in Wireless Sensor Networks," Sensors, vol. 12, no. 8, pp. 11113–11153, 2012.

[14] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in Advances in Neural Information Processing Systems (NeurIPS 2012), 2012, pp. 1097–
1105

[15] S. Madakam, R. Ramaswamy, and S. Tripathi, "Internet of Things (IoT): A Literature Review," Journal of Computer and Communications, vol. 3, no. 5, pp. 164–173, 2015.