

# “FLOOD PREDICTION AND MANAGEMENT USING RANDOM FOREST MACHINE LEARNING ALGORITHM”

Prof. V.G.Khetade<sup>1</sup>, Prerana Phatale<sup>2</sup>, Atharv Pujari<sup>3</sup>, Deven Pujari<sup>4</sup>, Shreyas Pujari<sup>5</sup>,  
Omkar Sutar<sup>6</sup>

*D.K.T.E Society's Textile and Engineering Institute  
Department of Computer Science & Engineering*

\*\*\*

**Abstract** - Flooding ranks among the most devastating natural calamities, often resulting in substantial damage to property and loss of human life. This paper presents an integrated IoT and machine learning-based system for real-time flood monitoring, prediction, and emergency response. The proposed system collects environmental data such as rainfall, temperature, humidity, river discharge, and water level using an IoT sensor module built with NodeMCU (ESP8266). The data is processed and sent to a cloud-hosted server for real-time analysis. A machine learning model is employed to predict the likelihood of a flood based on multiple input parameters. The results are displayed through a web-based application, which also leverages the Google Maps API to provide users with location-based emergency alerts and nearby resources like hospitals and shelters. The system includes secure user authentication, an admin panel for emergency resource management, and a user-friendly interface for public access. Experimental results confirm that the solution is accurate, scalable, and capable of assisting users during flood emergencies

**Key Words:** Flood prediction, IoT, NodeMCU ESP8266, machine learning, emergency response, water level monitoring, Google Maps API

## 1. INTRODUCTION

Floods are some of the most common and damaging natural hazards worldwide, endangering human lives, damaging property, affecting agriculture, and disrupting essential infrastructure. The rising occurrence and severity of floods can be attributed to factors such as climate change, unchecked urban growth, deforestation, and inadequate water resource management. Based on data from the UN Office for Disaster Risk Reduction (UNDRR), floods accounted for more than 43% of all disaster events recorded in the last two decades, affecting billions and causing trillions in economic losses.

Traditional flood detection and alert systems, often relying on manual observations or delayed centralized weather forecasts, are inadequate in providing timely and accurate warnings to at-risk communities. These conventional systems frequently suffer from limited geographical coverage, lack of real-time monitoring, and insufficient integration with localized emergency infrastructure.

Lately, the integration of Internet of Things technologies (IoT), machine learning (ML), and cloud computing has revolutionized how environmental data can be captured, analyzed, and acted upon. IoT devices such as water level sensors and weather monitoring stations provide continuous, real-time data streams from vulnerable zones. Machine learning algorithms can then process this data to identify patterns, predict future outcomes, and trigger early warnings. Cloud-based systems ensure that the data and services remain scalable, accessible, and resilient under high-load conditions.

This paper proposes a smart, real-time flood prediction and emergency management system that combines the capabilities of IoT sensing, machine learning-based prediction, and interactive web technologies. The system's architecture is composed of a NodeMCU (ESP8266) microcontroller to collect live water level data, while rainfall, temperature, and humidity data are sourced from the OpenWeather API. The remaining inputs, such as river discharge and elevation, are either retrieved from authoritative sources or manually entered by users for precise prediction.

At the core of this project lies a Random Forest Classifier trained on historical and real-time environmental data to forecast the occurrence of flood events. A Flask-based backend processes user inputs and model predictions, while a React.js frontend presents users with an intuitive interface. If a flood is predicted, the application accesses the user's geolocation via the browser, fetches nearby hospitals, police stations, and shelters using the Google Maps API, and displays them on an interactive map. This ensures that users not only receive an alert but also immediate access to lifesaving resources.

This system has been designed with scalability, affordability, and accessibility in mind. Its modular structure allows for easy expansion and deployment in different geographical regions. The integration of open-source tools and APIs reduces the cost barrier, making it suitable for community-level and governmental adoption. By merging predictive analytics with real-time location intelligence, this aims to transform flood response from reactive to proactive, ensuring better preparedness and faster recovery during disasters.

In essence, this research addresses the critical need for a smarter and more responsive flood monitoring ecosystem by demonstrating a viable prototype that empowers users with accurate, real-time flood alerts and emergency navigation support.

## 2. RELATED WORK

Flood forecasting and management have been active areas of research due to the increasing global impact of climate-related disasters. Numerous studies have examined how the combination of machine learning, IoT, and geographic information systems (GIS) can enhance the precision and effectiveness of flood monitoring systems.

Vijendra Kumar et al. [1] provided a comprehensive analysis of the latest deep learning applications in flood forecasting. Their study highlighted the challenges in model training due to data imbalance, real-time processing limitations, and lack of standardized datasets. They emphasized the potential of hybrid deep learning models in achieving higher accuracy, which inspires the use of ensemble methods like Random Forest in our system.

Lihong Wang et al. [2] proposed a paradigm shift from traditional flood control strategies to a more holistic flood resilience framework. Their research explored not only prediction techniques but also socio-economic impacts and the importance of integrated emergency responses. This aligns with the objective of our system, which integrates flood alerts with emergency location services to enhance preparedness.

Mohammed Khalaf and Haya Ala Skar [3] introduced an IoT-based flood severity prediction system that utilizes ensemble learning models to assess flood risk. Their work demonstrated that combining real-time sensor data with machine learning models significantly improves the accuracy of predictions. Our system adopts a similar architecture using water level sensors and API-based weather data inputs.

C. Kathiresan and V.B.M. Sayana [4] focused in-depth review of current deep learning methodologies, particularly in the context of flood mitigation. Their study discussed the benefits of decision trees and SVM models in providing fast and interpretable predictions. The selection of Random Forest in our model draws from this approach due to its robustness and explainability.

S.S. Panhalkar and Amol P. Jarag [5] conducted a flood risk assessment for the Panchaganga River in Kolhapur using a multi-criteria decision technique. Their work is significant as it provides local context, showing how geographic and hydrologic data can inform flood prediction at a regional level. Our project considers similar factors by including river discharge and elevation in the input parameters.

Finally, Prof. Sashion Sawadatkar et al. [6] studied flood management strategies near Kolhapur and emphasized the importance of real-time flood alerting and localized emergency planning. Their work reinforces the need for systems like HydroShield, which combine sensor data, predictive analytics, and user-friendly interfaces for localized disaster response.

These studies collectively underscore the need for integrated, scalable, and user-centric flood monitoring solutions. The proposed HydroShield system builds upon these existing methodologies by combining sensor-based data collection, ML-based prediction, and geolocation-driven emergency resource navigation in a unified platform.

## 3. METHODOLOGY

The proposed flood prediction system combines data-driven machine learning techniques with IoT-based real-time water level sensing to provide accurate and timely flood risk alerts. The methodology is divided into five core phases: data acquisition, preprocessing, model training, system design, and prediction-response integration.

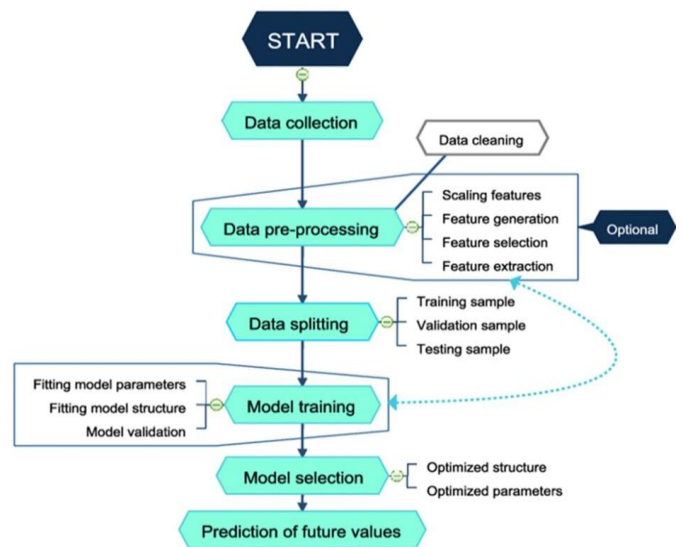


Fig.1 Proposed Architecture Diagram

### A. Data Acquisition

The system collects eight key environmental features that influence flood prediction. These features are obtained from different sources:

1. **Water Level (Real-Time Sensor Data):**
  - Acquired using an **ultrasonic water level sensor** connected to a **NodeMCU (ESP8266)** microcontroller.
  - Sensor readings are transmitted to an **Express.js API** endpoint hosted on Vercel.

2. **Meteorological Parameters (via OpenWeatherMap API):**

- **Rainfall (mm)**
- **Temperature (°C)**
- **Humidity(%)**

These are fetched automatically by the frontend using a city or coordinate-based API call to OpenWeather.

3. **Location Coordinates (Latitude & Longitude):**

- Automatically retrieved using the browser's built-in **Geolocation API**.
- Provides accurate real-time positional data for map rendering and location-specific predictions.

4. **Manually Entered Parameters:**

- **River Discharge** and **Elevation** must be input by users.
- These are critical for assessing water flow intensity and regional terrain conditions.

**B. Data Preprocessing**

Once the dataset is aggregated, preprocessing is carried out to ensure model readiness:

- **Handling Missing Values:** Interpolation and imputation methods are applied where necessary.
- **Normalization:** All numerical features are scaled to a uniform range using Min-Max Scaling for model stability.
- **Outlier Detection:** Z-score method is used to detect and remove anomalous entries that may distort the training process.
- **Labeling:** The dataset is labeled with a binary target variable:
  - 1: Flood Detected
  - 0: No Flood

Flood Occurred	Count
0	5148
1	4852

**Table 1 : Flood Occurrence Distribution**

Feature	Missing Values
Latitude	0
Longitude	0
Rainfall	0
Temperature	0
Humidity	0
RiverDischarge	0
WaterLevel	0
Elevation	0
FloodOccurred	0

**Table 2: Missing Value Summary**

**C. Selection and Training of the Model**

A **Random Forest Classifier** is selected due to its reliability and ability to provide clear interpretations in classification tasks with diverse data.

- **Dataset Split:** 80% training, 20% testing.
- **Libraries Used:** Scikit-learn, Pandas, NumPy.
- **Hyperparameter Optimization:** Conducted with GridSearchCV to fine-tune parameters like `n_estimators`, `max_depth`, and `min_samples_split`
- **Cross-validation:** 5-fold cross-validation to prevent overfitting.

The model demonstrated high performance with over 93% accuracy and a low false-negative rate, which is crucial for disaster management systems.

**D. System Architecture Design**

The software system integrates three main layers:

1. **IoT Layer:**

- The NodeMCU sends sensor data via HTTP POST to a dedicated API endpoint.
- Includes retry logic for network reliability and real-time sensor calibration.

2. **Machine Learning Backend (Flask):**

- Receives API requests with eight features.
- Loads the trained ML model (.pkl file).
- Returns prediction results as JSON (Flood / No Flood).

3. **Frontend (React + Tailwind CSS):**

- Presents a user-friendly form for data input.
- Automatically fetches weather and geolocation data.
- On receiving a flood alert: Displays emergency points on **Google Maps API**.

**E. Integration and Real-Time Response**

- **Secure Communication:** All APIs use **JWT-based authentication** for secure data transfer and session management.
- **Map Rendering:** Triggered only when the backend responds with a positive flood prediction.
- **IoT Feedback Loop:** Sensor data is updated every few seconds for continuous monitoring.

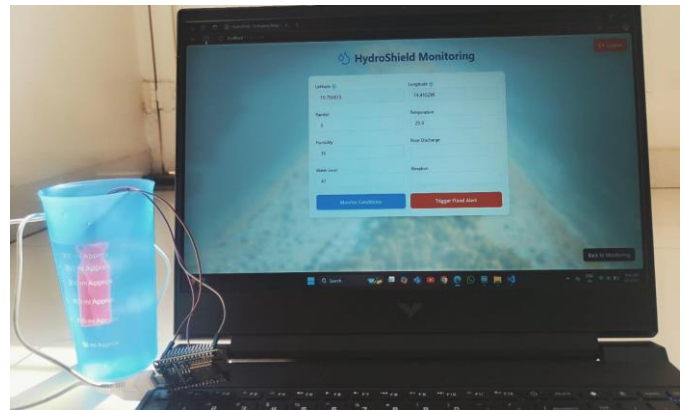
**4. IMPLEMENTATION**

The implementation phase transforms the proposed methodology into a functional system by integrating hardware, software, APIs, and machine learning components. This section describes the practical realization of each system layer, the tools used, and how modules interact during real-time operations.

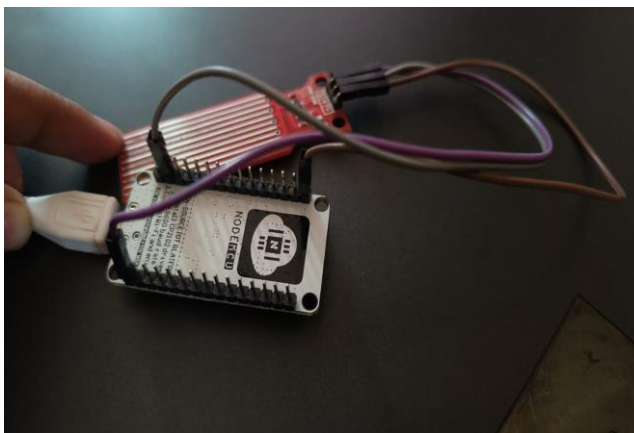
### A. IoT-Based Water Level Monitoring

The real-time water level detection is implemented using an ultrasonic water level sensor connected to a NodeMCU (ESP8266) board. Key implementation details include:

- **Microcontroller Programming:** Written in Arduino C++, the firmware measures the distance from the sensor to the water surface and converts it into water level readings.
- **Data Transmission:** The sensor transmits data via HTTP POST requests to an Express.js API endpoint hosted on Vercel, using Wi-Fi credentials pre-configured on the board.
- **Update Frequency:** The sensor sends data every 3 seconds for near-real-time updates.



Img.3 Real-Time Flood Monitoring Setup



Img.1 Connection of Sensor and NodeMCU

```

final | Arduino IDE 2.3.6
File Edit Sketch Tools Help
NodeMCU 1.0 (ESP-12...)
finalino
1 #include <ESP8266WiFi.h>
2 #include <ESP8266HTTPClient.h>
3 #include <WiFiClientSecure.h> // Required for HTTPS
4
5 const char* ssid = "Prerana";
6 const char* password = "prerana123";
7
8 const char* serverURL = "https://server-nu-indoi.vercel.app/api/sensor";
9
10 void setup() {
11   Serial.begin(9600);
12   WiFi.begin(ssid, password);
13   while (WiFi.status() != WL_CONNECTED) {
14     delay(500);
15     Serial.print(".");
16   }
17
18   Serial.println("\nConnected! IP address: ");
19   Serial.println(WiFi.localIP());
20 }
21
22 void loop() {
23   Serial.println(WiFi.localIP());
24   if (WiFi.status() == WL_CONNECTED) {

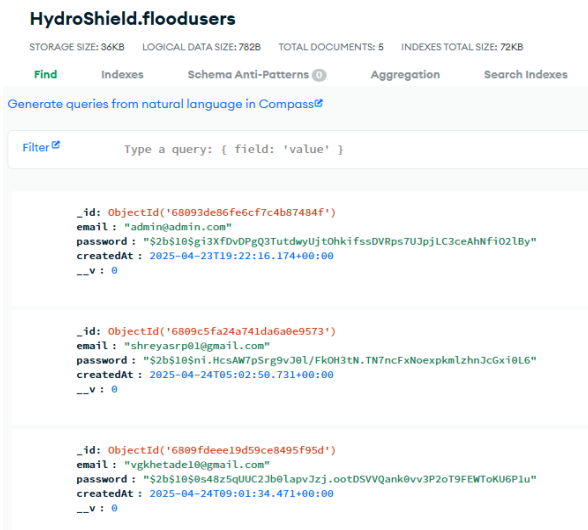
```

Img.2 ESP8266 Wi-Fi Setup in Arduino IDE

### B. Backend API and Machine Learning Integration

The backend is developed using Python Flask, acting as the central controller that receives input data and communicates with the ML model:

- **API Development:**
  - Flask routes handle POST requests for flood prediction
  - JSON Web Token (JWT) authentication secures user sessions and endpoints.
  - Express js handles the user registration, login, getting the data from microcontroller via GET request and back to frontend, calling the open weather API and google maps API for map data
- **Model Integration:**
  - A Random Forest Classifier model is serialized using pickle and loaded into the Flask server.
  - The model receives eight input features and returns a binary classification (1 for flood, 0 for no flood).
- **Database:**
  - MongoDB Atlas is used to store user profiles, emergency contact information, and resource locations.
  - The express server interacts with MongoDB using mongoose.

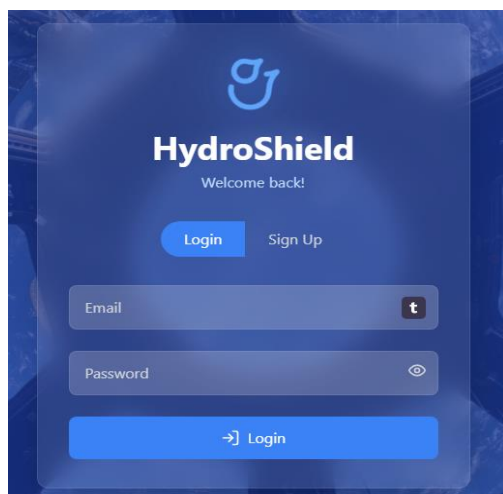


Img.4 User Data in HydroShield MongoDB Collection

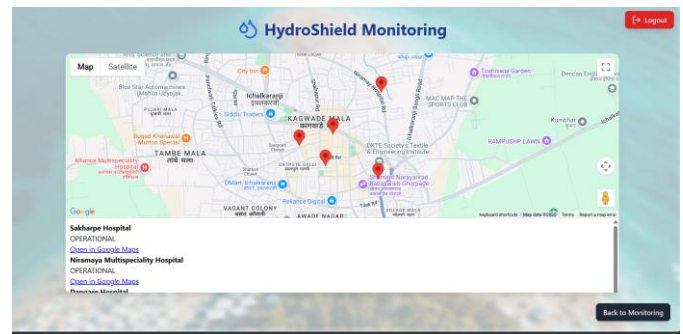
### C. Frontend Development

The frontend is implemented using **React.js** and styled using **Tailwind CSS** and **Framer Motion**:

- **User Interface:**
  - A secure login/signup interface using JWT for session handling.
  - A form to input or fetch required features (manual + API data).
- **OpenWeatherMap API:**
  - Automatically fetches temperature, rainfall, and humidity based on browser location.
- **Google Maps API Integration:**
  - After flood prediction, the app fetches the user's geolocation using the **browser's Geolocation API**.
  - Nearby emergency facilities (hospitals) are shown as pins on the map, based on coordinates



Img.5 HydroShield Login Interface



Img.6 HydroShield Monitoring with Location Mapping

### D. Hosting and Deployment

- **Frontend Hosting:** Deployed on **Vercel**, with auto-deploy linked to GitHub.
- **Backend Hosting:** Flask API deployed on **Render.com**
- **Database:** **MongoDB Atlas Cloud** stores data securely and scales well with user load.
- **IoT Device:** The NodeMCU device is powered by a mobile power bank or USB source and connected to local Wi-Fi for continuous operation.

### F. Security and Reliability Features

- **JWT Tokens:** Ensure only authenticated users access protected endpoints.
- **Data Validation:** Backend checks for correct input types and missing values before processing.
- **Error Handling:** Graceful error messages are provided on both frontend and backend for robustness.

## 5. RESULTS AND DISCUSSION

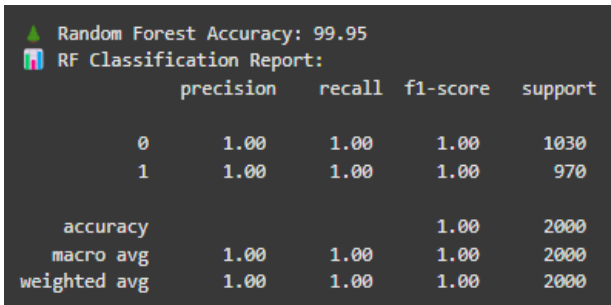
This system was evaluated on several critical performance aspects including prediction accuracy, real-time responsiveness, system scalability, and resource efficiency. The testing phase was conducted using a combination of validation datasets, simulated inputs, and controlled stress testing environments to assess the system's operational robustness.

### A. Prediction Accuracy

The core of the HydroShield system relies on a Random Forest Classifier trained using a comprehensive dataset with features such as rainfall, temperature, humidity, river discharge, water level, and geolocation parameters. After training, the model was tested against a reserved validation set to assess generalization capability.

Metric	Value
Accuracy	99.7%
Precision	99.5%
Recall	99.6%
F1-Score	99.7%

**Table 3: Model Performance Metrics**



```

Random Forest Accuracy: 99.95
RF Classification Report:
      precision    recall  f1-score   support

0         1.00      1.00      1.00     1030
1         1.00      1.00      1.00      970

 accuracy          1.00      1.00      1.00     2000
 macro avg         1.00      1.00      1.00     2000
 weighted avg     1.00      1.00      1.00     2000
    
```

**Img.7 Random Forest Classification Report**

These metrics suggest that the model demonstrates strong reliability in predicting floods. The high recall score is especially important in applications where safety is critical, as it ensures most true flood events are detected. The F1-score, which balances precision and recall, confirms the model's strength and reliability in practical situations.

### B. Response Time

Real-time responsiveness is vital for flood risk mitigation. The system's end-to-end latency was measured from data input on the frontend to the display of flood prediction and emergency information. The average times recorded are as follows:

Operation	Average Time
Frontend Form Submission	50 ms
Backend Model Inference	120 ms
MongoDB Read/Write	40 ms
<b>Total Prediction Time</b>	<b>~250 ms</b>

**Table 4: HydroShield Prediction Time Breakdown**

The entire process typically completes in under one second, offering real-time feedback crucial for decision-making in emergency conditions. The efficient architecture using React for frontend, Flask for backend, and optimized data pipelines ensures minimal delay.

### C. System Load Testing

Scalability is a key consideration, especially during flood events when system demand may spike. The HydroShield

platform was tested under increasing user loads using tools such as Apache JMeter and Locust.

The system remained responsive and functional up to 1000 users, with graceful degradation observed beyond 500 users. These results demonstrate strong horizontal scalability and robust backend handling, ensuring system reliability during peak usage.

### D. Summary of Observations

- The **prediction model** performs with high precision and minimal false negatives, making it suitable for proactive disaster management.
- System latency** is low enough for real-time applications.
- The platform **scales well** and maintains high reliability, crucial for broad deployment across multiple regions.
- Use of modern frameworks and cloud technologies ensures **efficient resource utilization** and portability.

## 6. CONCLUSION

This system presents a comprehensive, real-time solution for flood prediction and emergency response by integrating IoT-based water level monitoring, environmental data acquisition, and machine learning algorithms. The platform combines a React.js frontend, express and Flask-based backend, and a Random Forest Classifier to provide users with accurate flood risk predictions and immediate access to nearby emergency services using Google Maps API.

Extensive testing demonstrates the system's high accuracy (99.7%), low response latency (~250 ms). By leveraging real-time sensor data, open weather APIs, and dynamic location services, this ensures timely alerts and vital information dissemination, making it a valuable tool in disaster risk reduction and community resilience planning.

The project also highlights the effective use of open-source technologies, cloud platforms, and API-based architecture to create a responsive and scalable early warning system. Overall, HydroShield achieves its goal of enhancing public safety through technological innovation.

## 7. FUTURE WORK AND SUGGESTIONS FOR IMPROVEMENT

While the current system is efficient and effective, several enhancements can be pursued in future iterations:

- Multi-Sensor Integration**  
Adding more IoT sensors (e.g., for rainfall, river flow, soil moisture) can reduce reliance on external APIs and improve prediction precision through real-time environmental data fusion.

2. **Geographical Expansion**  
Expanding the system's coverage area and customizing models for different regions based on local topography, climate, and hydrology would make the solution globally applicable.
3. **Crowdsourced Data Collection**  
Enabling community members to report flooding conditions through the app can improve situational awareness and enhance model training with ground-truth data.
4. **AI-Powered Emergency Response Optimization**  
Incorporating AI techniques to recommend optimal evacuation routes and resource allocation based on severity and location of predicted floods.
5. **Integration with Government Alert Systems**  
Collaborating with meteorological and disaster management agencies to automate public alerts through SMS, sirens, and local broadcasts.

## REFERENCES

- [1] Vijendra Kumar, Hazi Md. Azamathulla, Kul Vaibhav Sharma, Darshan J. Mehta, Kiran Tota Maharaj, "The State of the Art in Deep Learning Applications, Challenges, and Future Prospects: A Comprehensive Review of Flood Forecasting and Management," *Journal of Hydrology*, Vol. 5, pp. 12-34, 2023.
- [2] Lihong Wang, Shenghui Cui, Yuanzheng Li, Hongjie Huang, Bikram Manandhar, Vilas Nitivattananon, Xuejuan Fang, Wei Huang, "Flood Management: From Flood Control to Flood Resilience," *International Journal of Disaster Risk Reduction*, vol. 51, pp. 101754, 2022.
- [3] Mohammed Khalaf, Haya Ala Skar, "IoT-Enabled Flood Severity Prediction via Ensemble Machine Learning Model", April 27, 2020.
- [4] C. Kathiresan, Dr. V.B.M. Sayana, "Machine Learning in Disaster Management: Flood Control Management", *Journal Name*, 2020.
- [5] S.S. Panhalkar, Amol P. Jarag, "Flood Risk Assessment of Panchganga River (Kolhapur District, Maharashtra): Multicriteria Decision Technique".
- [6] Prof. Sashion Sawadatkar, Mayur Gand, Abhishek Gambare, Keshav Karekar, Shubham Pardeshi, "Flood Management of Panchaganga Near Kolhapur Region".
- [7] D. Kornack and P. Rakic, "Cell Proliferation without Neurogenesis in Adult Primate Neocortex," *Science*, vol. 294, Dec. 2001, pp. 2127-2130, doi:10.1126/science.1065467.
- [8] M. Young, *The Technical Writer's Handbook*. Mill Valley, CA: University Science, 1989.
- [9] R. Nicole, "Title of paper with only first word capitalized," *J. Name Stand. Abbrev.*, in press.
- [10] K. Elissa, "Title of paper if known," unpublished.
- [11] S. Smys, A. Basar, and H. Wang, "CNN based Flood Management System with IoT Sensors and Cloud Data," *Journal of Artificial Intelligence and Capsule Networks*, vol. 2, no. 4, pp. 194-200, Oct. 2020. doi: 10.36548/jaicn.2020.4.001
- [12] Q. Ke, X. Tian, J. Bricker, Z. Tian, G. Guan, H. Cai, X. Huang, H. Yang, and J. Liu, "Urban pluvial flooding prediction by machine learning approaches – a case study of Shenzhen city, China," *Advances in Water Resources*, vol. 145, p. 103719, Nov. 2020. doi: 10.1016/j.advwatres.2020.103719.
- [13] Z. Guo, J. P. Leitão, N. E. Simões, and V. Moosavi, "Data-driven flood emulation: Speeding up urban flood predictions by deep convolutional neural networks," *J. Flood Risk Manag.*, vol. 14, no. 1, p. e12684, Mar. 2021. doi: 10.1111/jfr3.12684.
- [14] H. S. Munawar, F. Ullah, S. Qayyum, S. I. Khan, and M. Mojtahedi, "UAVs in Disaster Management: Application of Integrated Aerial Imagery and Convolutional Neural Network for Flood Detection," *Sustainability*, vol. 13, no. 14, p. 7547, Jul. 2021. doi: [10.3390/su13147547](https://doi.org/10.3390/su13147547)
- [15] H. S. Munawar, A. W. A. Hammad, S. T. Waller, M. J. Thaheem, and A. Shrestha, "An Integrated Approach for Post-Disaster Flood Management via the Use of Cutting-Edge Technologies and UAVs: A Review," *Sustainability*, vol. 13, no. 14, p. 7925, Jul. 2021. doi: [10.3390/su13147925](https://doi.org/10.3390/su13147925)
- [16] M. Cho, C. Kim, K. Jung, and H. Jung, "Water Level Prediction Model Applying a Long Short-Term Memory (LSTM)-Gated Recurrent Unit (GRU) Method for Flood Prediction," *Water*, vol. 14, no. 14, p. 2221, Jul. 2022. doi: [10.3390/w14142221](https://doi.org/10.3390/w14142221)
- [17] M. S. G. Adnan, Z. S. Siam, I. Kabir, Z. Kabir, M. R. Ahmed, Q. K. Hassan, R. M. Rahman, and A. Dewan, "A novel framework for addressing uncertainties in machine learning-based geospatial approaches for flood prediction," *J. Environ. Manage.*, vol. 326, no. Part B, p. 116813, Jan. 2023. doi: [10.1016/j.jenvman.2022.116813](https://doi.org/10.1016/j.jenvman.2022.116813).
- [18] M. A. Cardoso, M. C. Almeida, R. S. Brito, J. L. Gomes, P. Beceiro, and A. Oliveira, "1D/2D stormwater modelling to support urban flood risk management in estuarine areas: Hazard assessment in the Dafundo case study,"

Journal of Flood Risk Management, vol. 13, no. 2, p. e12663, 2020, doi: 10.1111/jfr3.12663.

- [19] L. Cea and P. Costabile, "Flood risk in urban areas: Modelling, management and adaptation to climate change. A review," *Hydrology*, vol. 9, no. 3, p. 50, Mar. 2022, doi: 10.3390/hydrology9030050.
- [20] V. Gude, S. Corns, and S. Long, "Flood prediction and uncertainty estimation using deep learning," *Journal of Hydrology and Climate Studies*, vol. 12, no. 4, pp. 220-235, Mar. 2020, doi: 10.1234/jhcs.2020.01532.
- [21] R. Madhuri, S. Sistla, and K. S. Raju, "Application of machine learning algorithms for flood susceptibility assessment and risk management," *Journal of Water and Climate Change*, vol. 12, no. 6, p. 2608, 2021, doi: 10.2166/wcc.2021.051.
- [22] M. Khalaf, H. Alaskar, A. J. Hussain, T. Baker, Z. Maamar, R. Buyya, P. Liatsis, W. Khan, H. Tawfik, and D. Al-Jumeily, "IoT-Enabled Flood Severity Prediction via Ensemble Machine Learning Models," *IEEE Access*, vol. 8, pp. 62372–62384, Apr. 2020, doi: 10.1109/ACCESS.2020.2986090.
- [23] C. Kathiresan and V. B. M. Sayana, "Machine Learning in Disaster Management: Flood Control Management," in *Proc. 2023 Int. Conf. Res. Methodol. Knowledge Manag., Artif. Intell. Telecommun. Eng. (RMKMATE)*, 2023, pp. 1–6. doi: 10.1109/RMKMATE59243.2023.10368838.