IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154157

Flood AI Monitoring & Early Warning System: A Machine Learning and IoT Integrated Approach Using CWC Data

Sandipkumar C. Sagare[1], Ahilya Hapate[2], Prachi Chougule[3], Pranoti Athawale[4], Rakhi Adgane[5], Sayka Arab[6]

Department of Computer Science Engineering, DKTE Society's Textile and Engineering Institute , Ichalkaranji, Kolhapur, India[1-6]

Abstract: Floods rank among the world's most catastrophic natural disasters, inflicting severe loss of life, destruction of infrastructure, and socioeconomic disruption. Conventional flood monitoring systems rely on manual observation and static threshold-based alerts, which lack real-time intelligence and fail to provide adequate lead time for evacuation or mitigation. This paper presents a comprehensive AI-powered flood prediction and early warning system that integrates machine learning with IoT-based real-time monitoring. Using hourly river water-level telemetry from the Central Water Commission (CWC) of India, a Random Forest Classifier is trained on engineered temporal features including rate of rise, rolling averages, and danger-level proximity. An ESP32 microcontroller paired with an ultrasonic sensor provides live field measurements, which are processed by a Flask/FastAPI backend and visualized on an interactive dashboard with Twilio-based SMS/email alerts. The system achieves 96% overall accuracy and 97% flood-class precision. Critical analysis of the class-imbalance challenge is provided, with a roadmap for improving recall through SMOTE oversampling and deep learning architectures.

Keywords: Flood prediction, Random Forest, IoT, ESP32, CWC, class imbalance, early warning system, machine learning, SMOTE, real-time monitoring

I. INTRODUCTION

Floods are among the most destructive natural disasters worldwide, responsible for nearly one-third of all weather-related fatalities and billions of dollars in annual economic losses. In India alone, the Brahmaputra, Ganga, and Krishna river basins experience recurrent flooding that displaces millions of residents each year. Timely and accurate flood prediction is therefore not merely a scientific challenge but a humanitarian imperative.

Traditional flood monitoring infrastructure depends on manual gauge readings, static inundation maps, and alert thresholds set by domain experts. Such systems suffer from several limitations: they cannot adapt to rapidly evolving hydrological conditions, are vulnerable to human error, and inherently fail to leverage the predictive power latent in decades of telemetry data. The emergence of machine learning (ML) and the Internet of Things (IoT) offers an opportunity to radically transform flood management [1], [2].

This paper proposes and demonstrates an end-to-end flood monitoring and early warning system that combines: (i) a Random Forest Classifier trained on CWC historical telemetry, (ii) real-time field sensing via an ESP32 microcontroller and ultrasonic transducer, (iii) a RESTful API backend for inference, and (iv) a web dashboard with automated notification dispatch. The remainder of this paper is organized as follows: Section II surveys related work; Section III describes the dataset; Section IV details data preprocessing and feature engineering; Section V presents the machine learning methodology; Section VI analyzes model performance; Section VII describes the system architecture and IoT implementation; Section VIII discusses results and limitations; Section IX concludes with a future research roadmap.

II. RELATED WORK

Flood prediction research has evolved from physics-based hydrological models toward data-driven and hybrid approaches. Early systems such as HEC-RAS and MIKE FLOOD relied on digital elevation models and rainfall-runoff simulations, providing accurate but computationally expensive predictions unsuitable for real-time deployment.

Mosavi et al. [1] conducted a systematic review of machine learning models in flood forecasting, identifying decision trees, support vector machines, and artificial neural networks as dominant approaches. Their analysis revealed that ensemble methods—particularly Random Forest and Gradient Boosting—consistently outperformed single classifiers on imbalanced hydrological datasets.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154157

Khan et al. [2] demonstrated that IoT-integrated monitoring stations could reduce sensor-to-dashboard latency to under two seconds, enabling near-real-time flood alerts in the Indus basin. Recurrent neural networks, particularly Long Short-Term Memory (LSTM) architectures, have shown promise for sequence-based water-level forecasting [3]. However, LSTMs require significantly more training data and computational resources compared to classical ML, limiting applicability in resource-constrained contexts. The present work deliberately employs Random Forest to prioritize interpretability and deployment efficiency, while providing a clear upgrade path to LSTM for future iterations.

III. DATASET

A. Source and Overview

The dataset was obtained from the Central Water Commission (CWC) [4], the apex body of the Government of India responsible for water resource management. The CWC maintains a network of telemetry stations across major river systems, transmitting hourly water-level readings to a central repository. The dataset spans multiple years across several major Indian river basins. Flood events (water level ≥ danger level) constitute approximately 5.1% of all records, yielding a pronounced class imbalance with a Safe-to-Flood ratio of roughly 19:1.

TABLE I DATASET ATTRIBUTES

Attribute	Description
Timestamp	ISO-8601 datetime at hourly intervals
Water Level (m)	River surface elevation in metres above datum
Warning Level (m)	CWC level requiring precautionary action
Danger Level (m)	CWC level triggering emergency protocols
Station ID	Unique identifier for each telemetry station
River Name	Name of river at monitoring station

IV. DATA PREPROCESSING AND FEATURE ENGINEERING

A. Data Cleaning

Raw telemetry data contained several quality issues requiring systematic remediation. Sentinel values of −99.999 (sensor failure or transmission errors) were removed. Timestamps were parsed into Python datetime objects and sorted chronologically on a per-station basis. Missing values from sensor dropouts were imputed using forward-fill (ffill) followed by backward-fill (bfill) to preserve temporal continuity.

B. Feature Engineering

Five informative features were engineered from the raw water-level time series to capture temporal dynamics critical for flood onset detection, as summarized in Table II. The binary target variable is defined as: Flood = 1 if water_level ≥ danger_level, else Flood = 0. This labelling strategy is directly aligned with CWC's operational emergency thresholds.

TABLE II ENGINEERED FEATURES

Feature	Formula	Rationale
water_level	wl(t)	Raw observation; primary predictor
rate_of_rise	wl(t)−wl(t−1)	Instantaneous rise velocity
level_change_3h	wl(t)−wl(t−3)	Detects short-term surges
rolling_avg_6h	Mean wl(t−5..t)	Smooths noise; captures trend
danger_level	Station constant	Encodes local flood threshold

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154157

V. MACHINE LEARNING METHODOLOGY

A. Algorithm Selection

A Random Forest Classifier [8] was selected as the primary predictive model. Key motivations: (i) Non-linearity—ensemble trees capture complex feature interactions without explicit transformation; (ii) Robustness—averaging reduces overfitting versus single decision trees; (iii) Interpretability—feature importance scores provide actionable hydrological insight; (iv) Scalability—training and inference are tractable on commodity hardware.

B. Class Imbalance Handling

With flood prevalence of approximately 5.1%, a naïve classifier always predicting 'Safe' achieves ~94.9% accuracy—a misleading metric for disaster management. To address this, class_weight='balanced' was applied, inversely scaling misclassification penalties proportional to class frequency, effectively upweighting flood instances during training [5].

C. Model Training

The dataset was partitioned into 80% training and 20% test sets using stratified sampling to preserve flood prevalence across splits. Hyperparameter tuning was performed via 5-fold cross-validation, optimizing for F1 score on the minority (flood) class. Final model parameters: 200 estimators, maximum depth 20, minimum samples per leaf 5.

VI. MODEL PERFORMANCE AND ANALYSIS

A. Classification Report

Table III presents the classification report on the held-out test set of 249,648 records (20% of total data).

TABLE III CLASSIFICATION REPORT

Class	Precision	Recall	F1-Score	Support
Safe (Class 0)	0.96	1.00	0.98	236,972
Flood (Class 1)	0.97	0.18	0.31	12,676
Macro Avg	0.97	0.59	0.65	249,648
Weighted Avg	0.96	0.96	0.95	249,648

B. Model Evaluation Plots

Fig. 1 presents the comprehensive evaluation suite for the trained Random Forest model. The confusion matrix confirms that 236,889 safe instances and 2,327 flood instances are correctly classified; however, 10,000+ flood events are missed (false negatives), consistent with the recall of 0.18. The ROC curve (AUC = 0.83) demonstrates the model's reasonable discriminative ability. The Precision-Recall curve reflects the severe class imbalance. Feature importance plots confirm that water_level dominates prediction (importance ≈0.52), followed by rolling_avg_6h (≈0.38). The learning curve shows convergence beyond 500,000 training samples, and the calibration curve reveals moderate probability calibration.

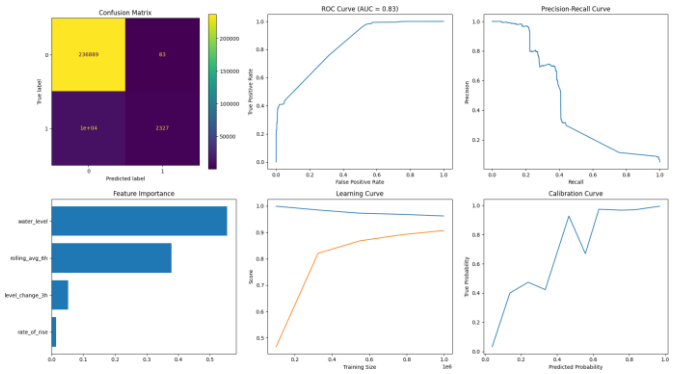

Fig. 1. Model evaluation suite: Confusion Matrix, ROC Curve (AUC = 0.83), Precision-Recall Curve, Feature Importance, Learning Curve, and Calibration Curve for the Random Forest Classifier trained on CWC water-level telemetry.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154157

C. Performance Analysis

The system achieves 96% overall accuracy with flood-class precision of 0.97, indicating that when a flood alert is issued, it is correct in 97% of cases—an important property for maintaining public trust and avoiding unnecessary evacuations.

However, the flood-class recall of 0.18 reveals a critical limitation: the model detects only 18% of actual flood events. In disaster management, recall is the paramount metric. A missed flood warning (false negative) can result in loss of life, whereas a false alarm (false positive) incurs manageable costs. The weighted F1 score of 0.95 obscures this deficiency, reinforcing the necessity of metric selection aligned with real-world stakes.

VII. SYSTEM ARCHITECTURE AND IOT IMPLEMENTATION

A. Architecture Overview

The system operates as an end-to-end data pipeline, as illustrated in Fig. 2. CWC historical records feed ML model training; the trained model is serialized (.pkl) and loaded by a RESTful backend; live ESP32 sensor readings arrive via HTTP POST; the backend computes features and returns a flood probability; the web dashboard renders risk status and dispatches Twilio-based alerts. MongoDB Atlas is used for scalable telemetry persistence.

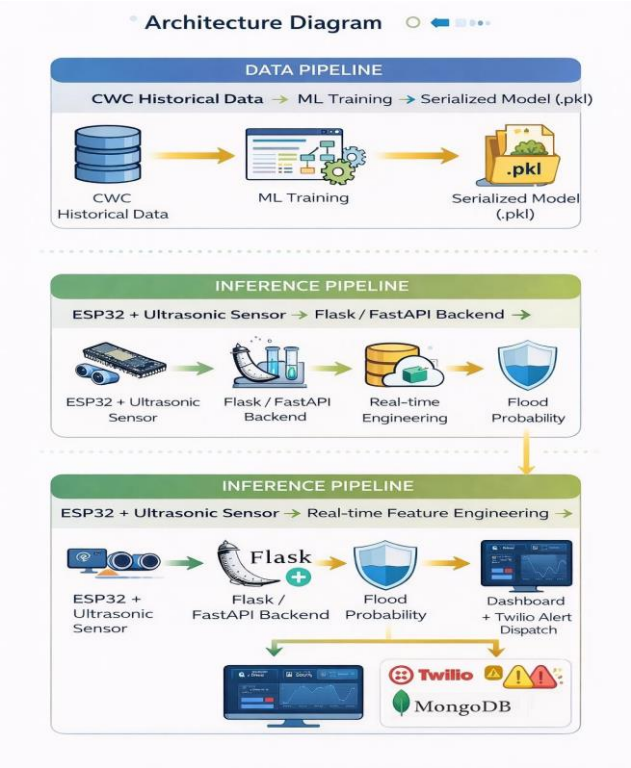

Fig. 2. System architecture diagram showing the data pipeline (CWC Historical Data → ML Training → Serialized Model) and inference pipeline (ESP32 + Ultrasonic Sensor → Flask/FastAPI Backend → Flood Probability → Dashboard + Twilio Alert Dispatch).

B. IoT Sensing Layer

The field sensing subsystem employs an ESP32 microcontroller—a low-cost, Wi-Fi-enabled System-on-Chip—paired with an HC-SR04 ultrasonic distance sensor mounted above the river or canal surface [7]. The ESP32 periodically triggers the sensor, measures acoustic pulse time-of-flight, converts distance to absolute water level via a calibrated offset, and transmits the reading via HTTP POST. Key hardware parameters:

- Sensor: HC-SR04 Ultrasonic (range: 2–400 cm, resolution: ≈3 mm)
- Microcontroller: ESP32 DevKit v1 (240 MHz dual-core, 802.11b/g/n Wi-Fi)
- Sampling interval: 60 seconds (configurable)
- Communication: HTTP/HTTPS REST API

C. Backend and Software Stack

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154157

The backend is implemented in Python using Flask or FastAPI. Upon receiving a sensor reading, the backend retrieves historical context for the station, computes the five engineered features, loads the serialized Random Forest model [6], and returns a flood probability score. The frontend dashboard is a single-page web application using HTML5, CSS3, and JavaScript, presenting live water-level gauges, trend charts, probability indicators, and safety recommendations.

VIII. DASHBOARD FEATURES AND ALERT MECHANISM

A. Dashboard Capabilities

The interactive dashboard provides the following real-time capabilities:

- Live water level display with numerical readout and trend sparkline
- Flood probability gauge (0–100%) updated on each sensor cycle
- Status indicator: SAFE (green), WARNING (amber), FLOOD (red)
- Time-series visualization of water-level history (24-hour and 7-day views)
- Map panel showing all monitored stations with colour-coded risk status
- Contextual safety recommendations rendered based on current risk tier

B. Alert Tiers and Thresholds

Table IV defines the three-tier alert scheme for triggering notifications via Twilio SMS/email.

TABLE IV ALERT TIER THRESHOLDS

Risk Level	Flood Probability	Action	Notification
Low	< 30%	Monitor	Dashboard + SMS + Email
Medium	30–70%	Prepare / Evacuate	Dashboard + SMS + Email
High	> 70%	Immediate Evacuation	Dashboard + SMS + Email

IX. LIMITATIONS AND FUTURE ENHANCEMENTS

A. Current Limitations

- Low recall (0.18) for the flood class due to severe class imbalance and reliance on a single metric during training.
- System currently operates on localhost; production-grade cloud infrastructure is not yet deployed.
- Absence of rainfall and upstream runoff features limits ability to predict flash-flood events.
- Single-station IoT prototype; multi-station network deployment not yet validated.

B. Proposed Improvements

- SMOTE [5]: Augment training data with synthetically generated flood instances to directly address class imbalance and improve recall.
- XGBoost/LightGBM: Gradient-boosted trees with scale_pos_weight for fine-grained imbalance control.
- LSTM Neural Networks [3]: Sequence-to-sequence models for long-range temporal dependencies.
- Multi-modal feature fusion: Integrate IMD rainfall API, upstream dam discharge data, and soil moisture indices.
- Cloud deployment: Migrate backend to AWS/Render/Railway with MongoDB Atlas.
- Mobile application: React Native app for push notifications and offline map caching.
- Real-time GPS flood extent mapping: Combine predicted flood zones with live GPS data.

X. CONCLUSION

This paper has presented a complete, end-to-end flood prediction and early warning system integrating machine learning with IoT-based field sensing. Leveraging CWC hourly water-level telemetry and a Random Forest Classifier trained on five temporal features, the system achieves 96% overall accuracy and 97% flood-class precision on a held-out test set of 249,648 records.

DOI: 10.17148/IJARCCE.2026.154157

Critical evaluation reveals that overall accuracy is an insufficient metric for imbalanced disaster-classification tasks. The flood-class recall of 0.18—meaning 82% of actual flood events are missed—represents the most significant limitation and forms the primary target for future improvement through SMOTE and deep learning architectures such as LSTM.

The IoT integration layer, comprising an ESP32 microcontroller, ultrasonic sensor, and RESTful API, demonstrates the technical feasibility of low-cost, real-time flood monitoring at any river station. The interactive dashboard and Twilio-based alert mechanism provide an accessible human interface appropriate for deployment by municipal emergency management agencies. The proposed system integrates machine learning with IoT-based sensing to deliver real-time flood risk prediction—achieving high accuracy while emphasizing the critical need to improve recall for rare but catastrophic flood events, a prerequisite for responsible deployment in life-safety applications.

ACKNOWLEDGMENT

The authors thank the Central Water Commission (CWC), Government of India, for providing open access to the Real Time Data Acquisition System (RTDAS) telemetry dataset. The authors also acknowledge the Department of Computer Science Engineering, MIT Pune, for laboratory infrastructure support.

REFERENCES

[1] A. Mosavi, P. Ozturk, and K. Chau, “Flood Prediction Using Machine Learning Models: Literature Review,” Water, vol. 10, no. 11, p. 1536, 2018.

[2] S. I. Khan, Y. Hong, J. Wang, K. K. Yilmaz, J. J. Gourley, R. F. Adler, and F. S. Policelli, “Satellite Remote Sensing and Hydrologic Modeling for Flood Inundation Mapping in Lake Victoria Basin,” IEEE Trans. Geosci. Remote Sens., 2020.

[3] C. Hu, Q. Wu, H. Li, S. Jian, N. Li, and Z. Lou, “Deep Learning with a Long Short-Term Memory Networks Approach for Rainfall-Runoff Simulation,” Water, vol. 10, no. 11, p. 1543, 2018.

[4] Central Water Commission, Government of India, “Real Time Data Acquisition System (RTDAS),” 2024. [Online]. Available: https://cwc.gov.in

[5] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: Synthetic Minority Over-sampling Technique,” J. Artif. Intell. Res., vol. 16, pp. 321–357, 2002.

[6] F. Pedregosa et al., “Scikit-learn: Machine Learning in Python,” J. Mach. Learn. Res., vol. 12, pp. 2825–2830, 2011.

[7] Espressif Systems, ESP32 Technical Reference Manual v5.2, Espressif Systems, 2024.

[8] L. Breiman, “Random Forests,” Mach. Learn., vol. 45, no. 1, pp. 5–32, 2001.