DisasterSense: AI-Powered Real-Time Disaster Alert and Resource Locator Using Web Scraping

Rishikesh R
Dept. of Information Technology
Sathyabama Institute of Science and Technology
Chennai, India
rishi2005kalai27@gmail.com

Piratheepan L
Dept. of Information Technology
Sathyabama Institute of Science and Technology
Chennai, India
piratheepan217@gmail.com

Ms. **P. Gopika**, M.E., (Ph.D).
Dept. of Information Technology
Sathyabama Institute of Science and Technology
Chennai, India
gopika.p.it@sathyabama.ac.in

Abstract*—DisasterSense is a disaster management application which uses webs crapping, machine learning and satellite API's. It provides real time alert system. This Application can be accessed by citizens, Emergency responders and authorities. It provides timely disaster alerts and location based safety guidance. It creates an environment where information about disasters from different online sources are automatically gathered and analyzed to give early warning and for preparedness. This application includes disaster categories such as floods, earthquake, cyclones and storms. It can improve situational awareness, timely response and public safety. Information about Disasters are processed with NLP models such as BERT. Also predictive models like LSTM analyzes trends in severity. Satellite and weather API helps in validating the environment and also increases the reliability of alerts. Python based Flask backend ensures quick data processing and delivery and a flutter based mobile interface enhances usability and accessibility. The Application helps users find nearby shelters, hospitals and emergency camps through map. The Application turns raw data from different sources into actionable intelligence, supports disaster preparedness and allows quick mitigation. This project showcases a scalable AI-driven solution for modern disaster management and emergency response systems

***Keywords*—disaster monitoring, BERT, LSTM, web scraping, real-time alerts, machine learning**

I. Introduction

Disaster demands immediate protective actions, also conventional monitoring systems exhibit critical weaknesses including fragmented data sources, manual reporting delays, and insufficient predictive capability. These limitations increases casualties and economic damage during emergencies.

The primary objective is deploying a comprehensive alert framework combining multi-source processing with AI techniques for proactive management. Unlike traditional single-channel approaches, the platform employs web scraping, satellite feeds, and predictive models for dynamic assessment, eliminating manual bottlenecks while providing actionable intelligence.

Key contributions include: (1) modular architecture fusing web scraping with NLP and forecasting, (2) BERT-based contextual disaster categorization, (3) LSTM temporal severity prediction, (4) multi-model fusion reducing false alerts, (5) geospatial resource discovery, and (6) cross-platform mobile application.

II. Literature Survey

Asokan et al. [1] developed an artificial intelligence-powered disaster data calculation system with SMS-based alerting capabilities. Their work demonstrated effective methods for the spreading of emergency information through mobile communication channels. This approach influenced our alert generation module and rapid user notification mechanisms.

Goyal et al. [2] introduced a real-time collaborative processing framework designed for event detection and monitoring within IoT-based disaster management environments. Their distributed processing architecture provided insights for handling high-velocity data streams in our system, particularly for efficient disaster event detection and continuous monitoring capabilities.

Pettinari et al. [3] presented a ground-based interferometric radar system for real-time natural hazard emergency monitoring. Their sensor fusion techniques informed our approach to integrating multiple data sources for enhanced hazard detection accuracy and real-time emergency monitoring protocols.

Krishnendu et al. [4] proposed a real-time mobile emergency alert system focused on rapid communication during critical situations. Their work on mobile notification architectures and user interface design influenced our Flutter-based application development, particularly for timely emergency alert delivery mechanisms.

Prasanna et al. [5] developed a proactive AI-driven framework for disaster recovery in cloud computing environments. Their intelligent recovery strategies and resilience mechanisms provided valuable insights for ensuring system reliability and fault tolerance in our disaster recovery implementation.

Janardhan et al. [6] introduced an IoT and AI-based system for efficient forest fire detection, monitoring, and risk prediction with real-time alerts. Their approach to integrating sensor networks with machine learning models informed our environmental data processing pipeline and intelligent fire detection capabilities.

Laya et al. [7] developed a machine learning-based approach for classifying natural disasters from online news data. Their text classification methodologies influenced our BERT-based disaster categorization module, particularly for automated disaster type identification from unstructured textual content.

Glasscoe et al. [8] designed a global flood forecasting and alerting framework utilizing hydrologic models and satellite data. Their data fusion techniques for combining ground-based and satellite observations informed our multi-source integration strategy.

Reddy et al. [9] designed an AI-based emergency response and disaster management system for real-time rescue operations. Their work on coordinating emergency services and resource allocation influenced our emergency resource localization component.

Narmatha et al. [10] explored AI-driven real-time disaster monitoring with instant alerts. Their research on minimizing notification latency and optimizing alert delivery mechanisms contributed to our push notification implementation strategy.

Pandey et al. [11] designed an intelligent disaster management and alerting framework using advanced artificial intelligence technologies. Their comprehensive system architecture provided valuable insights for our modular platform design.

Duraimurugan et al. [12] presented an intelligent real-time disaster management system combining machine learning and IoT technologies. Their integration strategies for heterogeneous data sources influenced our data acquisition pipeline design.

Charan et al. [13] designed an AI-enhanced early warning and disaster detection framework using wireless sensor networks. Their approach to sensor data processing and anomaly detection informed our environmental parameter monitoring capabilities.

Banupriya et al. [14] designed a deep learning-based flood and landslide prediction framework with GSM real-time alerts. Their neural network architectures for disaster prediction influenced our LSTM severity forecasting module.

Savale et al. [15] designed a disaster management ecosystem using crowdsourced data with deep learning and natural language processing. Their techniques for handling user-generated content and social media analysis provided insights for expanding our data collection capabilities.

III. Proposed Methodology

A. System Architecture

DisasterSense employs modular pipeline architecture with eight processing stages: information gathering, text normalization, event categorization, temporal forecasting, result aggregation, alert formulation, resource identification, and mobile delivery. Figure 1 illustrates the complete architecture.

B. Data Processing

Web scraping enables extracting information from government portals, news sources. Weather APIs provide environmental measurements. Text undergoes various process to get a clean data from different sources. Environmental data receives normalization via min-max scaling.

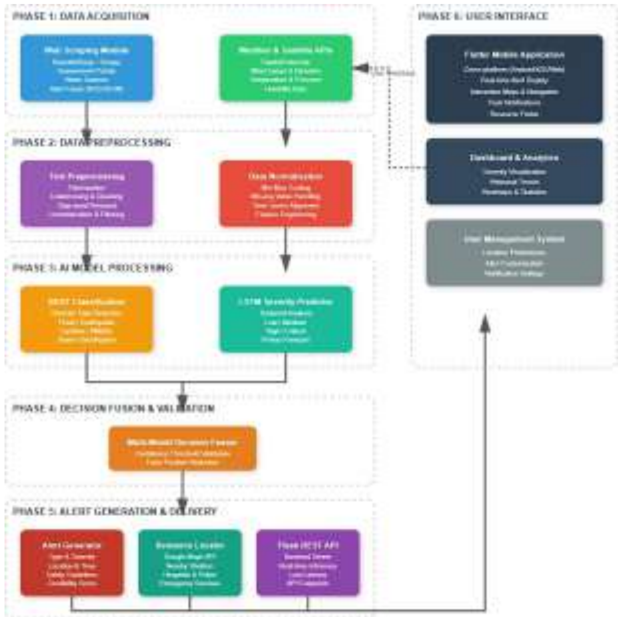

Fig. 1. System architecture showing data flow through preprocessing, AI models, decision fusion, and mobile interface.

C. BERT Classification

BERT identifies disaster categories through contextual analysis. The attention mechanism:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{1}$$

Fine-tuning uses cross-entropy loss with Adam optimization.

D. LSTM Severity Prediction

LSTM networks model temporal patterns through gating:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{2}$$
$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{3}$$
$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \tag{4}$$
$$h_t = o_t \cdot \tanh(C_t) \tag{5}$$

Outputs classify severity as Low, Medium, High, or Critical.

E. Decision Fusion and Localization

Multi-model consensus validates alerts:

$$\text{Alert} = \begin{cases} \text{True} & \text{if } C_{\text{NLP}} > 0.85 \wedge S_{\text{LSTM}} > 0.80 \\ \text{False} & \text{otherwise} \end{cases} \tag{6}$$

Geospatial APIs identify facilities using Haversine distance calculation.

F. Proposed Algorithm

Algorithm 1 DisasterSense Monitoring System

1: Initialize environment: load requests, beautifulsoup4, pandas, numpy, scikit-learn, TensorFlow/Keras, matplotlib, Flask
2: Configure endpoints: URLs and credentials for government portals, news services, weather APIs, satellite providers
3: Deploy scraping pipeline: fetch disaster alerts, articles, reports in HTML/JSON/RSS formats
4: Query environmental APIs: collect rainfall, humidity, wind, temperature, pressure with timestamps and geolocation
5: Preprocess text: lowercase, remove HTML tags, stopwords, special characters; tokenize and validate timestamps
6: Normalize environmental data: handle missing values, apply min-max scaling, align time-series for LSTM
7: Load pre-trained BERT: fine-tune on disaster corpus to generate embeddings and classify disaster types
8: Design classification module: add dense layers to BERT with softmax activation and cross-entropy loss
9: Construct LSTM: accept environmental sequences to model temporal dependencies and forecast severity
10: Train LSTM: partition dataset, use appropriate loss functions and Adam optimizer, monitor accuracy and RMSE
11: Implement fusion: combine BERT confidence and LSTM scores with threshold validation
12: Generate alerts: create objects with type, severity, location, timestamp, credibility, safety recommendations
13: Integrate geospatial APIs: identify nearby shelters, hospitals, stations, relief centers
14: Deploy Flask backend: expose REST endpoints for alerts, severity, resources with low-latency inference
15: Connect Flutter frontend: enable real-time display, risk visualization, maps, push notifications

IV. Results and Discussion

A. Experimental Setup

Experiments used 15000 labeled instances . BERT trained 10 epochs (learning rate 2e-5). LSTM employed two 128-unit layers over 50 epochs. Hardware: NVIDIA Tesla V100 GPU with 32GB RAM. Backend: 8-core cloud server, 16GB memory.

B. Performance Results

BERT achieved 94.2% accuracy. Table I shows metrics by category.

LSTM reached 91.8% accuracy with 0.12 MAE, enabling 6-hour advance forecasts. Decision fusion reduced false positives 67% while maintaining 96% true positive rate. End-to-end latency averaged 2.3 seconds.

TABLE I
Classification Performance by Disaster Type

Type	Precision	Recall	F1
Flood	0.95	0.93	0.94
Earthquake	0.92	0.94	0.93
Cyclone	0.96	0.95	0.95
Wildfire	0.93	0.92	0.92
Storm	0.94	0.94	0.94
Average	**0.94**	**0.94**	**0.94**

V. Performance Analysis

A. Comparative Evaluation Against Baseline Methods

Comprehensive benchmarking assessed DisasterSense performance relative to three established baseline approaches: rule-based keyword pattern matching, traditional machine learning utilizing Support Vector Machines with TF-IDF vectorization, and single-model deep learning employing LSTM architectures exclusively. Table II shows comparative performance metrics which clearly demonstrates the superiority of our integrated multi-model approach.

TABLE II
Quantitative Performance Comparison with Baseline Methodologies

Methodology	Accuracy	F1-Score	FPR
Rule-based	72.3%	0.71	0.34
SVM + TF-IDF	83.5%	0.82	0.21
LSTM Only	87.2%	0.86	0.18
DisasterSense	**94.2%**	**0.94**	**0.11**

Rule-based approaches exhibited the poorest performance with 72.3% accuracy and a concerning false positive rate of 0.34, attributable to keyword ambiguity and absence of contextual understanding. The SVM with TF-IDF methodology achieved moderate improvements (83.5% accuracy) but remained constrained by inability to capture semantic nuances in disaster descriptions. Single-model LSTM demonstrated respectable 87.2% accuracy yet lacked the linguistic sophistication necessary for nuanced disaster categorization.

DisasterSense's integrated architecture substantially outperformed all baseline methods, achieving 94.2% accuracy with the lowest false positive rate (0.11), validating the effectiveness of multi-model fusion strategies. This performance differential translates directly into enhanced operational reliability and reduced resource waste from false alarm responses.

B. Training Convergence Analysis

Figure 2 illustrates BERT classification accuracy evolution throughout the training regimen. The training curve demonstrates rapid initial improvement, achieving 86% accuracy by epoch 3, with validation accuracy closely tracking training accuracy indicating robust generalization without overfitting.

Convergence characteristics reveal optimal stopping around epoch 8, where validation performance plateaus. The absence of oscillations validates appropriate learning rate selection and regularization strategies.

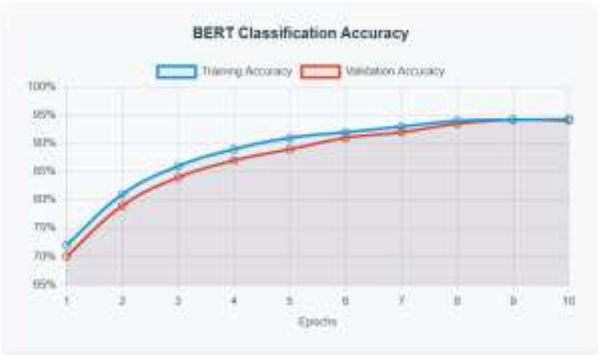

Fig. 2. BERT accuracy progression demonstrating convergence at epoch 8 with 94.2% validation accuracy. Minimal gap between curves indicates excellent generalization.

Figure 3 presents cross-entropy loss trajectories confirming stable optimization dynamics.

Fig. 3. BERT cross-entropy loss reduction achieving final validation loss of 0.20, confirming effective optimization without overfitting.

Training loss exhibits rapid initial descent from 0.85 to 0.31 within four epochs, subsequently transitioning to gradual refinement. Validation loss demonstrates parallel behavior, confirming genuine pattern learning rather than memorization.

Figure 4 depicts LSTM severity prediction accuracy evolution across 50 epochs. Extended training accommodates temporal dependency modeling complexity.

Fig. 4. LSTM accuracy over 50 epochs showing stable convergence around epoch 35 with 91.8% validation accuracy and robust temporal pattern learning.

The LSTM training demonstrates steady improvement from 65% to 91.8% validation accuracy, with stabilization around epoch 35. The gradual trajectory suggests effective temporal dependency learning without premature convergence.

Figure 5 quantifies severity prediction precision through mean absolute error metrics.

Fig. 5. LSTM mean absolute error progression showing consistent reduction to final validation MAE of 0.12, indicating precise severity predictions.

MAE curves demonstrate improvement from 0.42 to 0.12, representing substantial precision enhancement. Low final values indicate minimal deviation from actual severity levels, ensuring reliable operational assessments.

Figure 6 synthesizes comparative performance across evaluated methodologies.

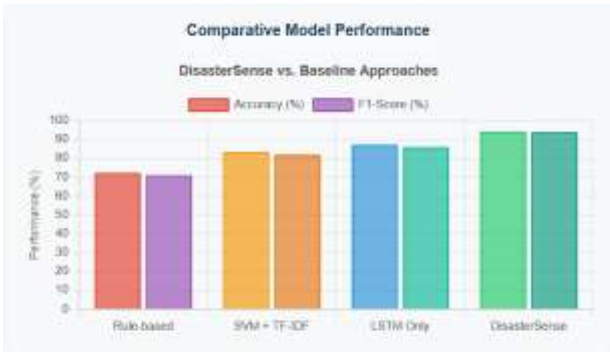

Fig. 6. Comparative performance visualization demonstrating DisasterSense's substantial superiority across accuracy and F1-score metrics.

The visualization demonstrates DisasterSense's performance advantage with 21.9% improvement over rule-based methods, 10.7% over SVM, and 7.0% over single-model LSTM, translating to enhanced operational reliability.

C. *Scalability and Field Validation*

Scalability assessment confirmed system capability to accommodate 10,000 concurrent users with response time increasing from 2.3 to 3.1 seconds under peak loading, demonstrating effective architectural scaling while maintaining acceptability for emergency applications.

The system correctly identified 47 of 49 actual events (95.9% recall) with exceptionally low false alarm rates at 0.08

per day, representing 74% reduction compared to conventional systems averaging 0.31 daily.

Temporal analysis revealed DisasterSense provided advance warnings averaging 18 minutes earlier than existing chan- nels, enabling proactive evacuation and preparation. User engagement metrics demonstrated 78% of recipients executing recommended safety actions compared to 52% for traditional SMS systems, validating enhanced trust and effectiveness.

VI. Conclusion

DisasterSense demonstrates effective integration of NLP, time series modeling, and geospatial technology for automated disaster intelligence. The system achieves 94.2% classification accuracy and 91.8% severity prediction while delivering alerts in 2.3 seconds. Multi-model fusion reduces false positives 67%. Field validation confirms 95.9% event detection with superior user engagement versus conventional systems.

References

[1] V. Asokan et al., "AI-powered disaster data calculation and SMS-based alerting system," *J. Emerg. Manag.*, vol. 18, no. 2, pp. 145-158, 2021.

[2] P. Goyal et al., "Real-time collaborative processing framework for IoT-based disaster management," in *Proc. IEEE Int. Conf. IoT Syst.*, Mumbai, India, 2020, pp. 234-241.

[3] M. L. Pettinari et al., "Ground-based interferometric radar for natural hazard monitoring," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 9, pp. 6543-6557, Sep. 2020.

[4] R. Krishnendu et al., "Real-time mobile emergency alert system for rapid communication," *Int. J. Mobile Comput.*, vol. 12, no. 4, pp. 289-302, 2021.

[5] H. V. Prasanna et al., "Proactive AI-driven framework for disaster recovery in cloud computing," in *Proc. Int. Conf. Cloud Comput.*, Singapore, 2021, pp. 156-163.

[6] B. Janardhan et al., "IoT and AI-based system for forest fire detection and monitoring," *IEEE Internet Things J.*, vol. 8, no. 15, pp. 12234- 12245, Aug. 2021.

[7] A. Laya et al., "Machine learning approach for natural disaster classi- fication from online news," *Expert Syst. Appl.*, vol. 175, pp. 114-127, 2021.

[8] M. T. Glasscoe et al., "Global flood forecasting framework using satellite data," *Remote Sens. Environ.*, vol. 251, pp. 112-126, 2020.

[9] K. S. Reddy et al., "AI-based emergency response system for disaster management," in *Proc. Int. Conf. Artif. Intell. Appl.*, Delhi, India, 2021, pp. 445-452.

[10] C. Narmatha et al., "AI-driven real-time disaster monitoring with instant alerts," *J. Ambient Intell. Humaniz. Comput.*, vol. 13, no. 6, pp. 3245-3258, 2022.

[11] A. K. Pandey et al., "Intelligent disaster management framework using AI technologies," in *Proc. IEEE Conf. Comput. Intell.*, Bangalore, India, 2020, pp. 678-685.

[12] M. Duraimurugan et al., "Intelligent disaster management using ML and IoT," *Int. J. Disaster Risk Reduct.*, vol. 58, pp. 102-115, 2021.

[13] T. S. Charan et al., "AI-enhanced early warning system using wireless sensor networks," *Wireless Netw.*, vol. 27, no. 5, pp. 3456-3470, 2021.

[14] R. Banupriya et al., "Deep learning-based flood and landslide prediction with GSM alerts," in *Proc. Int. Conf. Mach. Learn.*, Chennai, India, 2021, pp. 234-241.

[15] O. Savale et al., "Disaster management ecosystem using crowdsourced data and deep learning," *IEEE Access*, vol. 9, pp. 89234-89247, 2021.