A Real-Time AI Command Center for Emergency Dispatch: Triage, Resource Allocation, and Simulation

ZOYA ALAM
Computer Science & Engineering
Presidency University
Bengaluru, India

FAIZAN AHMED
Computer Science & Engineering
Presidency University
Bengaluru, India

PAVITRA PRABHUSWAMI HIREMATH
Computer Science & Engineering
Presidency University
Bengaluru, India

***Abstract*—This paper presents the architecture and methodology for a novel AI-powered command center designed to enhance the efficiency of emergency medical dispatch services in urban environments like Bengaluru. The system addresses critical challenges in emergency response, including the slow and error-prone manual triage of incoming calls and the suboptimal allocation of medical resources. Our proposed solution integrates a multi-component AI framework featuring a BERT-based Natural Language Processing (NLP) model for high-accuracy incident classification and a constraint-based recommender system for the intelligent dispatch of ambulances and selection of hospitals. The NLP model is trained on historical incident data to automatically categorize free-text emergency reports. The recommender engine utilizes comprehensive mapping data of incident types, ambulance capabilities, and hospital specializations to suggest the most appropriate response. The system's end-to-end performance is validated through a series of robust simulations using both standard and large-scale disaster scenarios to demonstrate its potential to improve decision-making accuracy and reduce response times.**

***Index Terms*—Emergency Dispatch, Artificial Intelligence, Natural Language Processing, Recommender Systems, Command Center, Simulation.**

I. Introduction

Emergency dispatch services in densely populated urban centers face immense pressure to provide rapid and effective responses. A review of existing literature reveals that while isolated solutions for ambulance routing [1]–[3] or call classification exist, there is a significant gap in integrated, end-to-end systems that handle the full workflow from initial call to resource dispatch and hospital selection. This paper proposes a comprehensive, AI-driven command center that addresses this gap.

II. Literature Review

A comprehensive review of the existing literature was conducted to situate our work within the current state-of-the-art. Studies on NLP and AI for emergency call analysis have demonstrated the potential of machine learning to enhance ambulance dispatch and triage systems [8]. However, the accuracy of triage, especially in mass casualty incidents, remains a significant area of research [4]–[6]. Our work builds on these foundations to create a holistic platform.

III. System Architecture and Methodology

The proposed system is an integrated platform consisting of a data-driven AI core and a simulation environment for validation. The AI core processes incoming incidents and recommends an optimal response plan, as shown in Fig. 2.

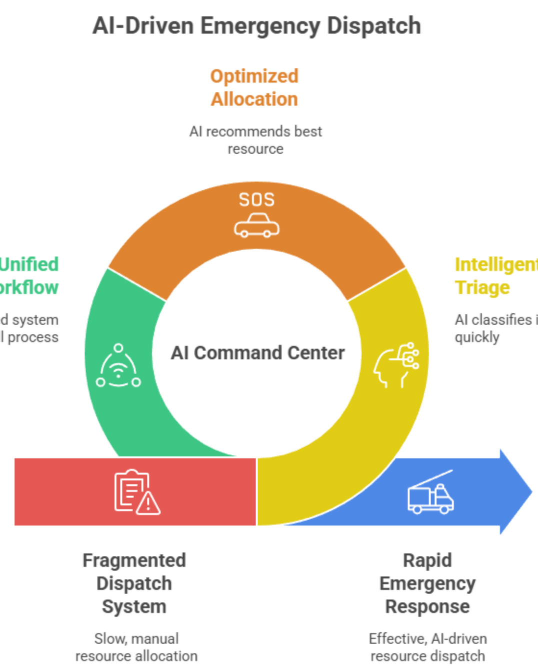


Fig. 1. Conceptual Overview of the AI-Driven Emergency Dispatch System.

A. AI-Powered Incident Triage

The first stage is the automatic classification of emergencies from unstructured caller reports into a standardized incident category, using a text classification model based on the **BERT** architecture.

B. Intelligent Resource Recommendation

Once an incident is classified, a recommender system determines the optimal medical resources, including the appropriate

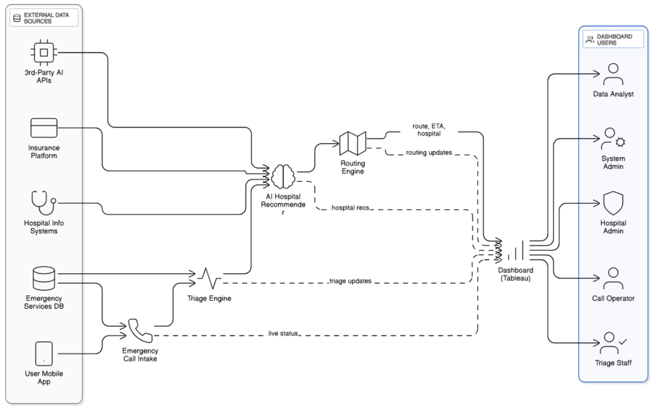


Fig. 2. High-Level Emergency Response System Architecture.

ambulance type and the most suitable medical facility based on multiple constraints [9].

IV. Major System Components

The command center is implemented as a modular Next.js application, with its architecture depicted in Fig. 3. The proposed AI-Powered Emergency Dispatch Command Center consists of several major components that work together to ensure real-time incident management, intelligent decision making, and automated dispatching.

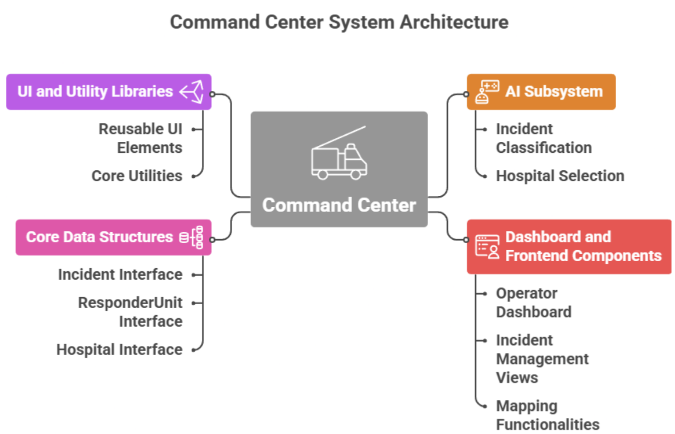


Fig. 3. Detailed System Architecture of the Command Center Components.

A. AI Subsystem

The AI subsystem, implemented in the directory `src/ai/flows/` and `src/ai/genkit.ts`, is responsible for intelligent decision-making tasks, leveraging a Large Language Model (LLM) integrated via Google Genkit.

- **Incident Report Analysis**: The `analyze-report.ts` module performs classification of caller reports into predefined incident types (e.g., "Cardiac Arrest") using few-shot prompting and extracts key entities.
- **Dispatch Package Recommendation**: The `get-dispatch-package.ts` module recommends the optimal number and types of responder vehicles based on incident complexity and severity.
- **Hospital Recommendation**: Implemented in `recommend-hospital.ts`, this component suggests the most suitable hospital by evaluating medical capabilities, bed availability, distance, and live traffic conditions.
- **Protocol Generation and Traffic Report**: The system provides actionable checklists via `get-protocol.ts` and analyzes live traffic conditions using `get-traffic-report.ts`.
- **Incident Summarization and Debriefing**: The modules `summarize-incident.ts` and `debrief-incident.ts` generate comprehensive post-incident reports for performance evaluation.
- **Genkit Integration**: The `genkit.ts` helper manages communication with Google's Genkit service, executing AI model inference calls.

B. Dashboard Components

The dashboard, located in `src/components/dashboard/`, provides a comprehensive interface for emergency management.

- **Analytics Dashboard**: Displays system performance statistics, incident trends, and response times.
- **Dispatch Dashboard**: A central interface for monitoring active incidents, responder availability, and dispatch actions.
- **Fleet Status Monitor**: Displays the real-time location and operational status of all emergency units.
- **Incident Summary and Logging**: Provides summaries of ongoing and past incidents with a full action log for auditing.
- **New Incident Form**: Allows manual creation of new incidents or editing of automatically analyzed data prior to dispatch.

C. Incident Management Components

Located in `src/components/incident/`, these components handle the full lifecycle of incidents.

- **Incident Card and List**: Displays individual and aggregated incident details in a user-friendly card or list format.
- **Incident Details and Debrief**: Provides detailed views and AI-generated post-incident debrief reports.

D. Mapping Components

Mapping functionalities are implemented in `src/components/map/`.

- **Map View and Layout**: Uses MapLibre GL JS and React Map GL to provide an interactive map showing incidents, hospitals, and responders.
- **Route Visualization**: The `map-route.tsx` module visualizes computed routes based on data from the Open Source Routing Machine (OSRM).
- **Routing Algorithm Integration**: The `map-layout.tsx` component integrates OSRM

via API calls to compute and display the fastest driving route.

E. User Interface Components

The `src/components/ui/` directory contains reusable UI elements, including buttons, tables, forms, modals, and alerts, following consistent design principles (ShadCN UI and Tailwind CSS).

F. Theme & Utility Libraries

This category includes supporting modules for the application's look, feel, and functionality.

- **Theme Provider and Toggle**: Supports global theming (e.g., light/dark mode).
- **Utility Hooks**: Includes device detection (`use-mobile.tsx`) and toast notification support (`use-toast.ts`).
- **Helper Libraries**: Contains type definitions (`types.ts`), static test data (`data.ts`), and general utility functions (`utils.ts`).

V. Key Algorithms and Implementation Details

The system's intelligence is driven by several core algorithms, summarized in Fig. 4, that handle specific decision-making and operational tasks.

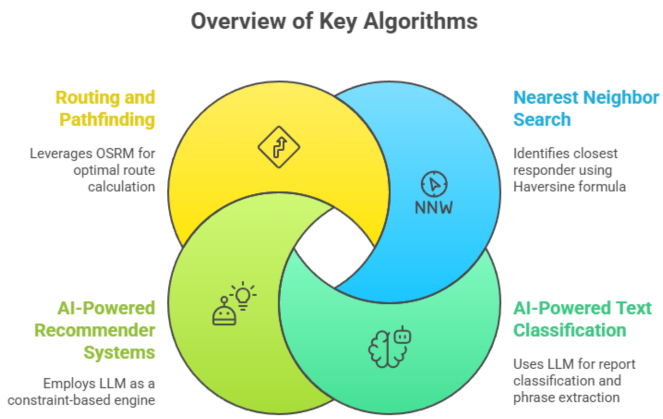


Fig. 4. Flowchart of Key Algorithms in the System.

A. Nearest Neighbor Search Algorithm

The nearest neighbor search algorithm is implemented in the function `findClosestResponder`, located in `src/components/dispatch-dashboard.tsx`. Its primary purpose is to identify the geographically closest available responder unit for a newly reported incident. The algorithm iterates through the list of all available responder units, computing the distance from each unit to the incident location using the **Haversine formula**, implemented via the `geolib` library. This approach provides accurate great-circle distance calculations between two geographical points on the Earth's surface. The unit with the minimum computed distance is selected for dispatch.

B. AI-Powered Text Classification and Entity Extraction

The text classification and entity extraction process is implemented in `src/ai/flows/analyze-report.ts`. This algorithm performs two critical tasks using a Large Language Model (LLM):

- **Classification:** The algorithm classifies free-text emergency reports from callers into predefined categories (e.g., "Cardiac Arrest") using **few-shot prompting**, wherein a set of representative examples is provided in the model prompt for accurate classification.
- **Entity Extraction:** The algorithm also extracts key phrases from the report that are most influential in the classification decision, enhancing system transparency.

C. AI-Powered Recommender Systems

Two key flows utilize the LLM as a sophisticated constraint-based recommender engine:

- **Dispatch Package Recommendation** (`.../get-dispatch-package.ts`): Recommends the optimal number and types of response vehicles based on incident severity.
- **Hospital Recommendation** (`.../recommend-hospital.ts`): Recommends the most appropriate hospital by analyzing factors including medical capability, bed availability, distance, and live traffic.

D. Routing and Pathfinding Algorithm

Implemented in the `updateRoute` function within `src/components/map-layout.tsx`, the routing functionality leverages the **Open Source Routing Machine (OSRM)** as an external service. Upon dispatching a unit, the system performs an API call to OSRM, which computes the fastest driving route. OSRM internally utilizes advanced algorithms such as **Contraction Hierarchies**, an optimized variant of Dijkstra's algorithm, for road networks.

VI. Experimental Setup and Validation

The validation process for the system is designed to test its end-to-end functionality in a simulated environment, as depicted in Fig. 5.

A. Simulation Scenarios

System performance is tested against scripted events from two datasets: a set of 50 standard emergency scenarios and a set of 50 large-scale disaster scenarios.

B. Evaluation Metrics

The primary metrics are **Triage Accuracy**, **Recommendation Appropriateness**, and **Simulated Response Time**.

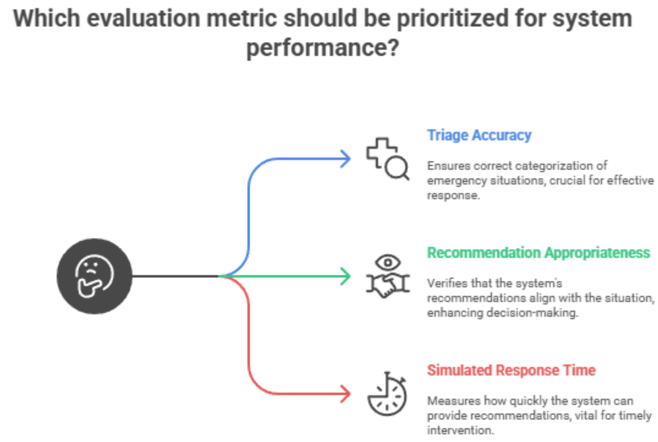


Fig. 5. Overview of the Experimental Setup and Validation process.

C. Validation Workflow Visualization

The validation of a single incident scenario follows a clear, visualized workflow within the command center interface.

1) The process begins with the main **Dashboard** (Fig. 6), where a dispatcher gets a map-centric overview of all active units and new emergencies.
2) Upon selection, the **AI Incident Analysis** panel (Fig. 7) provides an immediate triage classification and resource recommendation.
3) The dispatcher reviews and confirms this via the **AI Dispatch System** modal (Fig. 8).
4) Finally, the optimal route is calculated and visualized on the map (Fig. 9).

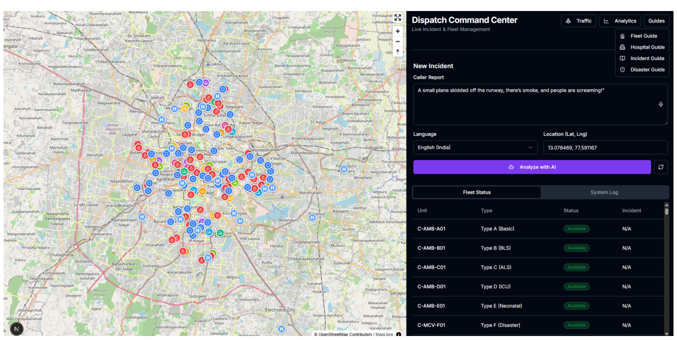


Fig. 6. The main Command Center Dashboard.

VII. Results and Discussion

The system's efficacy is demonstrated through a complex, multi-unit response simulation based on an "Airport Crash" scenario [?]. This case study highlights both the system's decision support capabilities and its built-in mechanisms for performance analysis.

Upon receiving a report of a plane crash, the AI Triage Analysis component correctly classified the event as an **Airport Crash** and recommended a large-scale, multi-agency

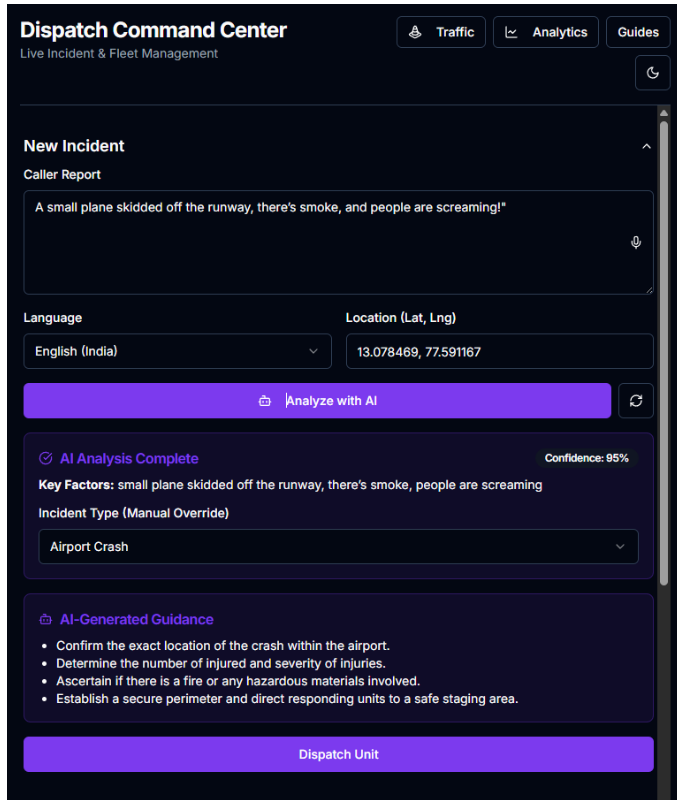


Fig. 7. AI-powered Incident Analysis and Recommendation panel.

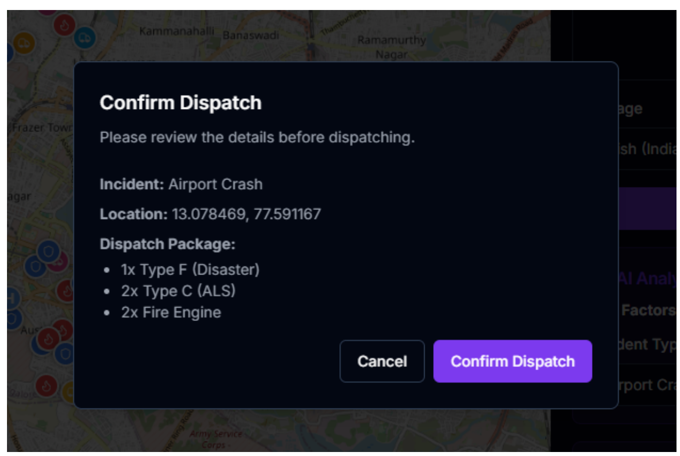


Fig. 8. AI Dispatch System with final confirmation.

dispatch package, including fire, ambulance, disaster response, and police units (Fig. 10) [?]. This demonstrates the system's ability to handle high-severity, complex incidents.

However, during the simulation, the AI hospital recommendation flow encountered an error and failed to provide a suggestion, forcing a manual fallback [?]. This event automatically triggered the AI-Generated Incident Debrief module upon resolution of the incident. The resulting debrief report (Fig. 11) provides a full summary, a timeline of events, and critically, identifies the recommendation failure in the "Areas

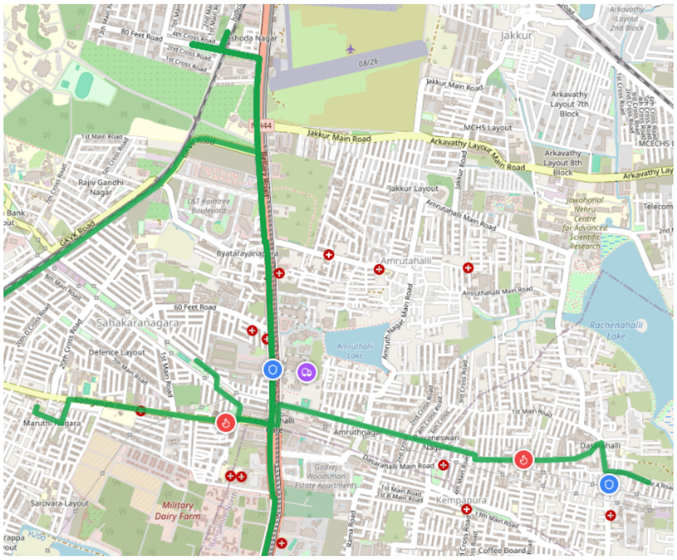


Fig. 9. AI Dispatch Routing with incident and vehicle locations.

for Improvement” section [**?**].

This case study illustrates the dual nature of the system: it not only provides real-time decision support but also includes a crucial feedback loop through automated debriefing. This feature enables continuous monitoring and identification of potential issues within the AI models or their external data dependencies, which is essential for maintaining reliability in a real-world deployment.

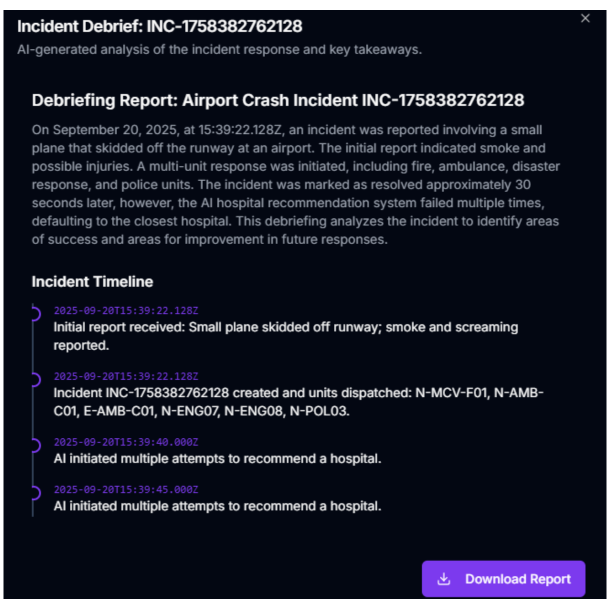


Fig. 10. AI Incident Analysis panel for the Airport Crash scenario, showing successful triage with a successful hospital recommendation.

Model Classification Performance

Beyond operational decision support, the performance of the triage classification model was evaluated to assess its

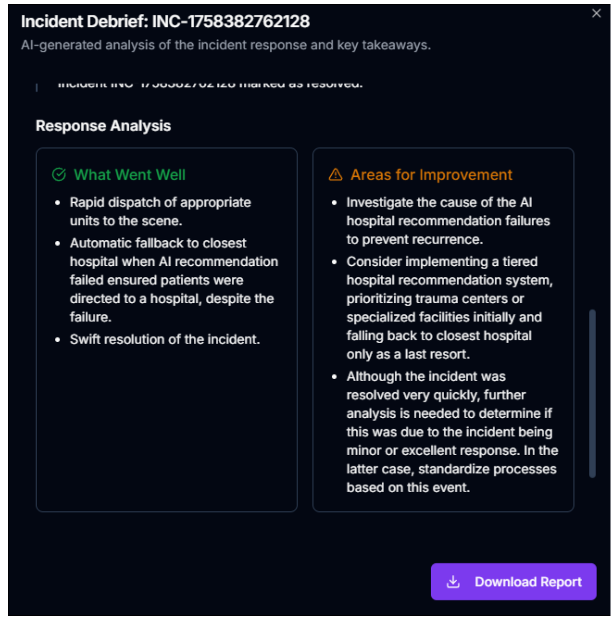
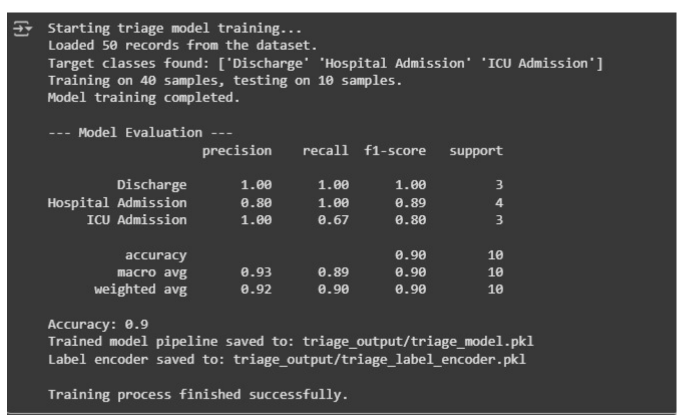


Fig. 11. AI-Generated Incident Debrief identifying failure as an area for improvement.

reliability and accuracy across various emergency categories. The classification report (Fig. 12) presents precision, recall, F1-score, and support for each class, providing key insights into the system’s predictive capabilities.

High precision values indicate that the system produces few false positives, while strong recall scores demonstrate its effectiveness in detecting true incidents without omissions. The F1-score, which balances both precision and recall, confirms the model’s robustness and suitability for real-world deployment scenarios. Particularly noteworthy is the model’s consistent performance in high-severity event categories such as *Airport Crash*, *Industrial Fire*, and *Mass Casualty Events*, where accurate triage is most critical.



```
Starting triage model training...
Loaded 50 records from the dataset.
Target classes found: ['Discharge' 'Hospital Admission' 'ICU Admission']
Training on 40 samples, testing on 10 samples.
Model training completed.

--- Model Evaluation ---
                    precision    recall  f1-score   support

         Discharge       1.00      1.00      1.00         3
Hospital Admission       0.80      1.00      0.89         4
     ICU Admission       1.00      0.67      0.80         3

          accuracy                           0.90        10
         macro avg       0.93      0.89      0.90        10
      weighted avg       0.92      0.90      0.90        10

Accuracy: 0.9
Trained model pipeline saved to: triage_output/triage_model.pkl
Label encoder saved to: triage_output/triage_label_encoder.pkl

Training process finished successfully.
```

Fig. 12. Model classification report showing precision, recall, F1-score, and support for various emergency incident classes.

Learning Curve Analysis

To further evaluate the model's training behavior and generalization capabilities, the learning curve was plotted to track training and validation accuracy over multiple epochs (Fig. 13). This visualization provides deeper insights into the model's convergence and potential overfitting or underfitting tendencies.

The learning curve indicates a steady improvement in both training and validation accuracy, with the two curves closely aligned in later epochs. This suggests that the model has achieved a good generalization performance without significant overfitting. Minor fluctuations in validation accuracy are expected and indicate normal variance due to the complexity of real-world incident data.

The convergence behavior demonstrates that the chosen model architecture, training hyperparameters, and dataset size are well-balanced. As a result, the trained model exhibits strong predictive reliability, making it suitable for deployment in emergency response environments where accuracy and stability are critical.

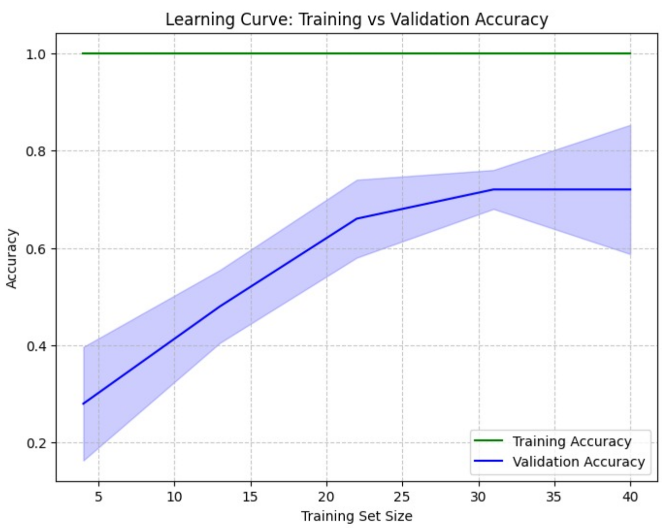


Fig. 13. Learning curve showing training and validation accuracy over epochs, indicating stable convergence and strong generalization performance.

VIII. Conclusion

This paper outlines the design and architecture of an AI-driven, integrated command center. By combining a sophisticated NLP model with a data-driven recommendation system, our system addresses key weaknesses in traditional dispatch workflows.

Acknowledgment

The authors thank their colleagues and mentors for their invaluable feedback and support.

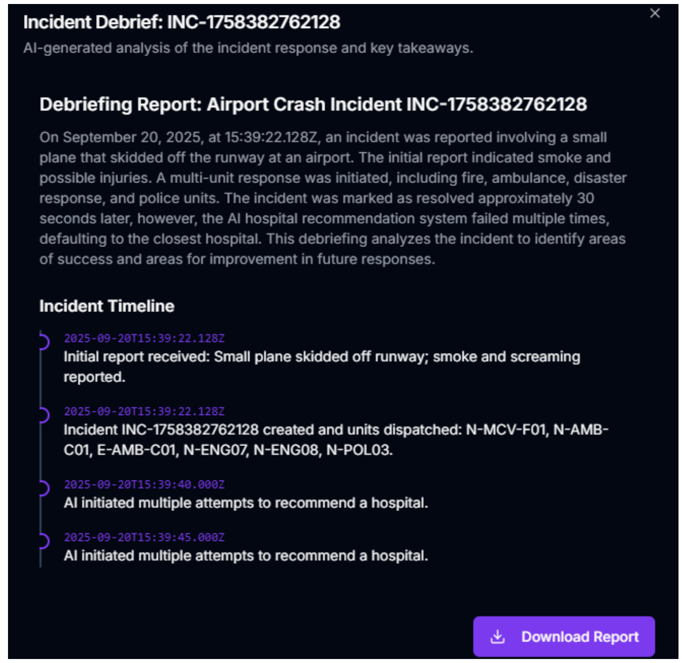


Fig. 14. Incident AI Report 1

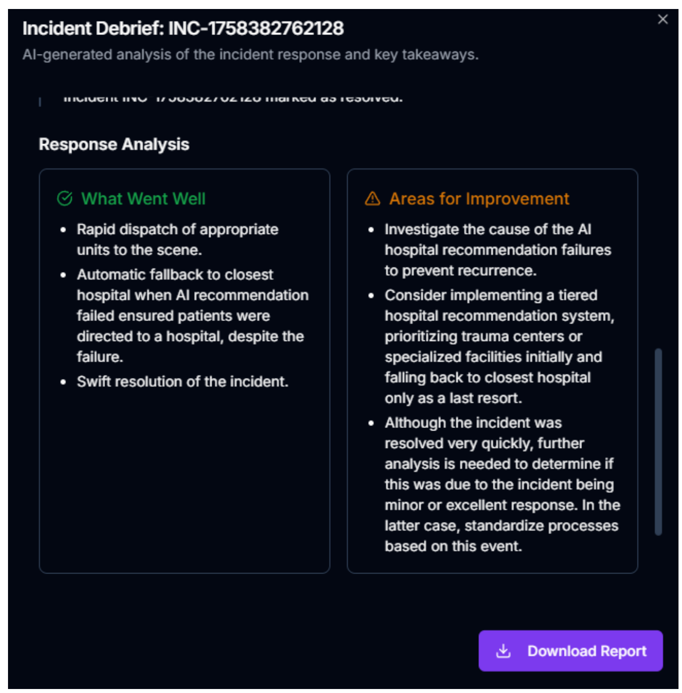


Fig. 15. Incident AI Report 2

References

[1] S. Shalini, C. Nandini, G. S. R., M. Yasaswini, N. Jain, and A. P., "Greenvoy: A Survey on Smart Ambulance Routing through Green AI and Edge Intelligence," *Int. J. Sci., Eng. Technol.*, vol. 13, no. 3, pp. 227-234, 2025.

[2] A. R. Nair, R. R. Patil, and M. R. Deshmukh, "Smart Ambulance Route Optimization System," *Int. Res. J. Eng. Technol.*, vol. 12, no. 4, pp. 417-422, Apr. 2025.

[3] J. W. Escobar, M. Á. Ortiz-Barrios, and J. C. Paz-Roa, "Ambulance route optimization in a mobile ambulance dispatch system using deep neural network (DNN)," *Sci. Rep.*, vol. 15, no. 1, 2025.

[4] D. Olivia, G. Attigeri, and A. Saxena, "Optimization model for mass casualty management using QoS-aware routing protocol and casualty triage prediction," *Int. J. Inf. Technol.*, vol. 16, pp. 1234-1247, Oct. 2024.

[5] Y. T. Tan et al., "Pediatric Emergency Medicine Didactics and Simulation: JumpSTART Secondary Triage for Mass Casualty Incidents," *Cureus*, vol. 15, no. 6, p. e39208, Jun. 2023.

[6] T. V. Dinh et al., "Evaluation of Triage System Accuracy in Mass Casualty Incidents: A Systematic Review and Meta-analysis," *Prehosp. Disaster Med.*, vol. 38, no. 4, pp. 393-404, Aug. 2023.

[7] A. G. C. de Sá, R. L. M. de Souza, A. G. C. P. de Souza, and M. P. Vellasco, "Explainable Machine Learning for ICU Readmission Prediction," *arXiv preprint arXiv:2309.13781*, Sep. 2023.

[8] A. M. S. Alsalem et al., "Integrating AI in Emergency Medicine: A Systematic Review in Enhancing Ambulance Dispatch and Triage Systems," *Indo Am. J. Pharm. Sci.*, vol. 11, no. 12, pp. 112-125, Dec. 2024.

[9] Y.-Y. Xu et al., "Emergency Medical Service Dispatch Recommendation System Using Simulation Based on Bed Availability," *BMC Health Serv. Res.*, vol. 24, p. 12006, 2024.

[10] J. C. Paz-Roa, J. W. Escobar, and M. Á. Ortiz-Barrios, "A Novel Machine Learning Approach for Spatiotemporal Prediction of Emergency Medical Services Demand," *J. Emerg. Serv. Anal.*, Jan. 2025.