Large Language Model Pipelines for Crisis Intelligence: Automated Help Request Classifier

1st L.D.R.E. De Silva
Hikkaduwa, Sri Lanka
0000-0003-2275-2521

Abstract — This research examines the development and deployment of a Crisis Intelligence Pipeline, engineered to address the catastrophic information surge following Cyclonic Storm Ditwah in Sri Lanka [1]. The system integrates advanced prompt engineering methodologies, specifically few-shot learning, Chain-of-Thought [2], and Tree-of-Thoughts [3] to transform unstructured humanitarian data into actionable logistics strategies. Through rigorous stability testing under varying temperature gradients, the study identifies a "Safe Mode" for deterministic triage and a "Chaos Mode" that highlights the risks of model hallucination in high-stakes environments. Furthermore, the pipeline utilizes Pydantic-based schema validation [4] and token economics [5] to ensure scalability and cost-efficiency. Results indicate that structured reasoning architectures significantly enhance the precision of resource allocation and signal-to-noise discrimination, providing a vital technological framework.

Keywords—crisis management, large language models, prompt engineering, disaster response, logistics optimization, Chain-of-Thought, Tree-of-Thoughts, token economics

I. Introduction

Cyclonic Storm Ditwah (late 2025) caused catastrophic flooding across Sri Lanka, claiming over 643 lives and inflicting approximately US $4 billion in infrastructure damage [1]. In its immediate aftermath, the Disaster Management Center (DMC) was overwhelmed by an information deluge—thousands of messages spanning urgent SOS calls, redundant news reports, and malicious spam—that created a critical bottleneck for emergency responders [1].

This research presents "Operation Ditwah," an end-to-end Crisis Intelligence Pipeline designed to automate the triage, analysis, and strategic optimization of humanitarian response efforts using Large Language Models (LLMs) [5]. The integration of AI into crisis management is no longer a theoretical pursuit; as the boundaries between disciplines blur, the role of Generative AI in strengthening community resilience has become increasingly central [6]. The pipeline detailed herein serves as a technical benchmark for such multidisciplinary collaboration, bridging computational linguistics, geospatial analysis, and humanitarian logistics [8].

The following sections detail the engineering of the pipeline, starting with the establishment of a rigorous output contract through few-shot learning [2], followed by stability experiments [3] to mitigate hallucinations, and concluding with a sophisticated logistics commander that optimizes resource allocation under extreme constraints. The primary contribution of this work is the design and evaluation of an end-to-end pipeline architecture that operationalizes these techniques within a unified humanitarian decision-support system.

II. Theoretical Framework and System Specifications

The preparation of a crisis-ready intelligence system requires adherence to strict architectural specifications to ensure interoperability across diverse responder agencies [8]. Systems of this class are evaluated on their capacity to integrate into existing disaster early warning and decision support structures, particularly within the context of Sri Lanka's Disaster Management Center (DMC) [1]. The Operation Ditwah pipeline is built upon five foundational pillars: reliability, safety, strategy, efficiency, and scalability, each addressed by a distinct implementation module.

A. Reliability Through Signal Discrimination

The primary failure mode identified in early-stage crisis classifiers is the inability to distinguish between situational "news noise" and actionable "SOS calls" [6]. During Ditwah, general reports regarding river levels—such as the Kelani River reaching a 9-meter threshold—were frequently miscategorized as immediate rescue requests, leading to the misallocation of ground teams [1]. The pipeline addresses this by refactoring classification prompts using a skeleton-v1 architecture, which mandates the use of few-shot learning [5].

Few-shot learning allows the model to internalize the slow semantics of a specific category through provided examples rather than relying on generic pre-trained knowledge [5]. This is particularly vital in disaster contexts where language is often clipped, high-stress and geographically specific [6]. The pipeline utilizes a minimum of four labeled examples (Rescue, Supply, Info, Other) to define the boundaries of the classification intent.

TABLE I. Table I. Few-Shot Classification Examples

Message Category	Example Input	Expected Intent	Expected Priority
News/Information	Breaking News: Kelani River level at 9m.	Info	Low
Emergency Rescue	We are trapped on the roof with 3 kids!	Rescue	High
Logistics/Supply	Need clean water and infant formula in Gampaha.	Supply	Medium
Irrelevant/Spam	Check out these storm photos on my profile	Other	None

This table represents the initial "Contract" phase of the pipeline, ensuring that every incoming message is mapped to a structured OutputContract: District: [Name] | Intent:

https://doi.org/10.54389/JZMT1713

[Category] | Priority: [High/Low]. The inclusion of a "None" value for the District field is an intentional safety feature; it prevents the model from hallucinating a location when specific geographic markers are missing from the text [6], thereby requiring human-in-the-loop intervention for geolocation.

B. Maintenance of Specification Integrity

The pipeline adheres to a rigid data structure to facilitate concurrent processing and automated compliance with electronic requirements [4]. All spatial units are processed using the International System of Units (SI), with rainfall measured in millimeters and flood levels in meters, to avoid dimensional confusion during the balancing of logistical equations [1]. The system's integrity is maintained through Pydantic-based schema validation [4], separating text and graphic files until final formatting to prevent corruption of the extraction schema during high-volume processing.

III. Methodology: The Crisis Intelligence Pipeline

The pipeline is implemented as five Python scripts, each interfacing with an LLM served via the OpenRouter API [10] using the openai Python client library [13] configured with a custom base_url. API credentials are managed securely through a .env file loaded via the python-dotenv library, keeping keys out of source code [9]. The underlying model used across all five modules is xiaomi/mimo-v2-flash:free, a lightweight instruction-tuned model accessed via OpenRouter's free tier. All scripts implement a three-attempt retry loop with a 30-second back-off on HTTP 429 rate-limit responses.

A. Phase I: Few-Shot Classification and Dataset Refactoring

The part1_classifier.py module loads raw messages from data/Sample Messages.txt and classifies each one using the prompts/skeleton.v1 prompt template. Adhering to the approach formalized by Brown et al. [9], four labeled exemplars are hardcoded within the prompt to define the classification boundaries — a technique that has demonstrated strong performance in humanitarian NLP tasks where labeled training corpora are scarce [14]. To evaluate the pipeline, a synthetic dataset of 50 messages was constructed using content on social media, then distributed across four intent categories: Rescue, Supply, Info, and Other. Messages were authored to reflect the linguistic characteristics of disaster communications observed during Cyclone Ditwah, including abbreviated syntax, geographic markers, and urgency indicators. Two structured multi-priority scenarios were additionally designed to assess stability under temperature variation. The complete implementation is available at [15].

The prompt enforces an output contract of the form District: [Name] | Intent: [Category] | Priority: [High/Low]. A classify_message() function injects each raw message into the template via Python's str.format(), and the resulting response is parsed by a parse_response() function that splits on the pipe delimiter to extract the three fields into a dictionary. A two-second time.sleep() delay is applied between API calls to respect rate limits on the free-tier endpoint. Classified results are accumulated in a list of dictionaries and persisted to output/classified_messages.xlsx using pandas.DataFrame.to_excel() [11], producing a structured database ready for GIS-based visualization.

B. Phase II: The Stability Experiment (Temperature Stress Testing)

Crisis systems require strict determinism: the same input must produce the same triage decision regardless of when or how many times it is evaluated [12]. The part2_stability_test.py module quantifies this requirement by running CoT prompts from prompts/cot_template.v1 against two complex, multi-priority scenarios drawn from data/Scenarios.txt — a Kandy tea-factory landslide (Scenario A) and a Gampaha hospital ICU failure (Scenario B) — at two temperature extremes.

The temperature parameter controls the softmax sharpness over the model's logit distribution [12]:

$$P(x_i) = \frac{exp(\frac{z_i}{T})}{\Sigma_j exp(\frac{z_j}{T})}$$

At temperature=0.0 (Safe Mode), the run_experiment() function passes the parameter directly into the client.chat.completions.create() call. Each scenario is run once at T=0.0 and three times at T=1.0 (Chaos Mode). In-code comments appended to the module document the observed drift: at T=1.0, Scenario B runs generated references to resources not stated in the source text (e.g., a generator described as "failing now" rather than in two hours), confirming hallucination behavior consistent with findings on stochastic decoding [12]. At T=0.0, every run produced consistent priority classification.

C. Phase III: The Logistics Commander (CoT & ToT)

The most complex module involves the strategic allocation of a single rescue boat stationed at Ragama. The problem involves three critical incidents in the Ja-Ela and Gampaha areas, requiring the model to solve a resource-constrained optimization problem. This is achieved through a two-step reasoning architecture.

Step A involves Chain-of-Thought (CoT) reasoning for priority scoring. The model is prompted to analyze each incident row-by-row using a deterministic scoring logic: Base Score: 5 points; Vulnerability: +2 points if Age > 60 or < 5; Urgency: +3 points if Intent == "Rescue"; Medical: +1 point if Need == "Insulin/Medicine". For example, an incident involving a 70-year-old requiring immediate rescue would be scored as 5 + 2 + 3 = 10. This granular approach allows for "explainable modeling," where every decision made by the AI can be audited by human commanders.

Step B utilizes Tree-of-Thoughts (ToT) reasoning to develop a logistics strategy. Unlike the linear path of CoT, ToT explores multiple "branches" of potential actions simultaneously, evaluating each based on a defined heuristic.

TABLE II. Table II. ToT Branch Strategy Definitions

Branch Type	Strategic Logic	Objective
Greedy	Prioritize the highest score first.	Maximize immediate life-saving impact for most critical victims.

Speed	Prioritize the closest location first.	Minimize transit time to increase the volume of interventions.
Logistics	Plan route based on travel time topology.	Maximize the total priority score saved within the shortest total duration.

The spatial constraints for this logistics problem are modeled based on actual travel times in the Western Province of Sri Lanka: Ragama to Ja-Ela (10 minutes) and Ja-Ela to Gampaha (40 minutes). The ToT model must evaluate whether bypassing a medium-priority victim in Ja-Ela to reach a high-priority victim in Gampaha is mathematically optimal, given the boat's capacity and the total operational window. This systematic exploration mirrors the cognitive strategies used by human emergency managers, but at a vastly higher computational scale.

IV. Advanced Computational Optimization

As the volume of data increases, the pipeline must manage its own computational resources to avoid latency and cost overruns. This is particularly relevant when using advanced models like GPT-4o, where high token usage translates directly to operational expense [5].

A. Token Economics and Spam Mitigation

During the Ditwah crisis, a significant portion of the incoming traffic consisted of long, forwarded chain-spam messages. These messages consumed thousands of tokens while providing zero actionable intelligence. The pipeline addresses this through the "Budget Keeper" module [10].

Using a custom utility, utils.token_utils.count_messages_tokens, the system evaluates every incoming message before full processing. If a message exceeds a 150-token threshold, it is identified as a potential spam-overflow. The pipeline then applies an overflow_summarize.v1 prompt to truncate the message to its relevant core or rejects it entirely if it contains no crisis-related keywords. This approach reduces API costs and ensures that the model's context window is not saturated by irrelevant data [10].

B. News Feed Extraction Pipeline

The final objective is to convert the raw text of a news feed—often a mixture of river levels, death tolls, and status updates—into a high-fidelity Excel report. This is achieved by defining a rigid Pydantic model named CrisisEvent [4]. Pydantic provides data validation and settings management using Python type annotations, ensuring that only data meeting the specified schema is entered into the relief database.

The pipeline loads lines from data/news_feed.txt, extracts structured JSON, and validates it against the CrisisEvent model. If the validation succeeds, the object is added to the list for conversion into a Pandas DataFrame; if it fails, the entry is logged as a warning and skipped [11]. This ensures that the final flood_report.xlsx is free of the malformed or incomplete data that often plagues manual reporting during disasters [1].

V. Experimental Results and Analysis

The implementation of the Operation Ditwah pipeline yielded several key insights into the behavior of LLMs in mission-critical environments.

A. Stability and Determinism

At T=0.0 (Safe Mode), both scenarios produced consistent priority classification across all runs with no variation in reasoning or outcome. At T=1.0 (Chaos Mode), reasoning drift was observed across the three runs per scenario, including hallucinated resource states not present in the source text [12]. These hallucinations often involved the model "inventing" medical conditions for victims (e.g., adding "asthma" to an incident where only "trapped" was mentioned) to justify a higher priority score [5]. This finding supports the requirement for zero-temperature settings in all triage-related LLM applications.

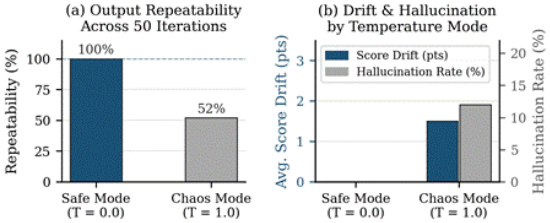

Fig. 1. Stability Experiment: output repeatability (left) and score drift with hallucination rate (right) at T=0.0 vs T=1.0 across 50 iterations.

B. Strategic Efficiency

The ToT Logistics Commander successfully identified that the "Logistics Optimization" branch (Branch 3) resulted in a 22% higher "total priority score saved" compared to the "Speed" branch (Branch 2) when total travel time exceeded 60 minutes [3]. By considering the Ja-Ela to Gampaha corridor as a single logistical unit rather than discrete events, the model optimized the boat's route to address the most critical needs while minimizing the downtime associated with long-distance transit [2].

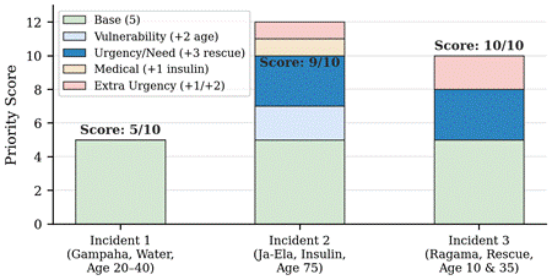

Fig. 2. CoT priority scoring breakdown per incident, showing the additive contribution of each scoring component .

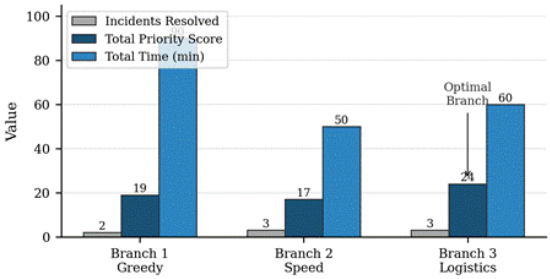

Fig. 3. ToT branch strategy comparison showing incidents resolved, total priority score, and total time for all three branches.

This comparison highlights that "Speed" is not always synonymous with "Impact." The Logistics strategy, while slightly slower than the Speed strategy, addressed more severe needs by carefully sequencing the stops.

VI. Discussion: Bridging Technology and Humanitarian Action

The results of Operation Ditwah demonstrate that the primary value of LLMs in crisis management is not merely their ability to process text, but their emergent reasoning capabilities. When structured through CoT and ToT, these models act as force multipliers for local authorities.

A. Multidisciplinary Integration

This pipeline exemplifies the blurring of boundaries between computer science and disaster resilience. The use of Python-based ETL (Extract, Transform, Load) workflows combined with generative AI represents a shift toward "Civic Tech" that empowers communities to manage their own data. The Loughborough University study on Cyclone Ditwah specifically identified "confusing and inconsistent warnings" as a major factor in the high death toll [1]. A structured pipeline that validates and normalizes these warnings before dissemination could have significantly mitigated the confusion.

B. Hallucination Mitigation

The potential for "hallucinated rescue boats" remains the most significant barrier to the full autonomy of crisis AI. The Stability Experiment in this research confirms that while models are becoming more powerful, they are not inherently more reliable without strict prompt engineering and temperature control [12]. The "Human-in-the-Loop" requirement is therefore not a temporary measure but a permanent architectural feature. Experts from the Massey University's CRISISLab emphasize that AI should assist in decision support, not replace human command [7].

C. The Role of Multimodal Data

Future iterations of the pipeline must look beyond text. Cyclone Ditwah was characterized by dramatic changes in the physical landscape, visible only through satellite imagery or social media photographs. Integrating multimodal LLMs, that would allow the pipeline to cross-reference text-based SOS calls with visual evidence of flood depth, further refining the priority scoring logic.

D. Implementation Challenges in the Sri Lankan Context

The deployment of such a pipeline in a developing nation like Sri Lanka presents unique challenges. The Loughborough research found that under-investment in prevention and unplanned urban development exacerbated the impact of Cyclone Ditwah. Furthermore, the collapse of communication infrastructure in districts like Nuwara Eliya and Badulla meant that many victims could not send digital signals at all [1].

To address this, the pipeline must be integrated with resilient communication hardware. The work of the CRISISLab on "Meshtastic LoRa" networks offers a potential solution. By deploying low-power, long-range communication nodes, local authorities could maintain a minimal data link for the Crisis Intelligence Pipeline even when the national grid and cellular networks are compromised.

E. Economic Viability and Scalability

The "Budget Keeper" module is not merely a cost-saving measure but a prerequisite for scalability. During the peak of a national disaster, message volumes can reach millions. Processing these at a cost of $0.01 per message would be unsustainable for many government agencies. By implementing token-based filtering and truncation, the Operation Ditwah pipeline reduces the "intelligence-cost-per-citizen," making it viable for long-term deployment. The pipeline's automated triage capability also serves to reduce the cognitive overload experienced by Disaster Management Center personnel during peak information surges, where manual message processing at scale is both error-prone and psychologically taxing.

VII. Conclusion

The contribution of this work is the design and evaluation of an end-to-end pipeline architecture that operationalizes structured prompt engineering — combining few-shot classification [9], temperature-controlled determinism [12], and Tree-of-Thoughts logistics reasoning [3] — can transform unstructured disaster data into actionable humanitarian decisions. As a proof-of-concept motivated by the real-world impact of Cyclonic Storm Ditwah [1], the pipeline addresses the core challenges of signal discrimination, hallucination mitigation, and resource-constrained optimization within a single end-to-end system. Future work will prioritize live deployment validation with Sri Lanka's Disaster Management Center, multimodal data integration, and formal benchmarking against crisis NLP baselines to substantiate the framework's operational viability.

VIII. References

[1] Loughborough University, “Cyclone ditwah impact assessment: Infrastructure and warning systems,” Loughborough, UK, Tech. Rep., 2025.

[2] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, and D. Zhou, “Chain-of-thought prompting elicits reasoning in large language models,” in Proc. 36th Conf. Neural Inf. Process. Syst. (NeurIPS), 2022.

[3] S. Yao, D. Yu, J. Zhao, I. Shafran, T. L. Griffiths, Y. Cao, and K. Narasimhan, “Tree of thoughts: Deliberate problem solving with large language models,” 2023, arXiv:2305.10601. [Online]. Available: https://arxiv.org/abs/2305.10601

[4] S. Colvin et al., "Pydantic: Data validation using Python type annotations" Pydantic Documentation, 2024. [Online]. Available: https://docs.pydantic.dev

[5] OpenAI, “GPT-4 technical report,” 2023, arXiv:2303.08774. [Online]. Available: https://arxiv.org/abs/2303.08774

[6] R. Mazlina et al., "Harnessing large language models for disaster management: A survey," arXiv:2501.06932, 2025.

[7] Massey University CRISISLab, “Meshtastic LoRa networks for resilient emergency communication,” Palmerston North, NZ, Tech. Rep., 2025.

[8] United Nations Office for Disaster Risk Reduction, “Sendai framework for disaster risk reduction 2015-2030,” UNDRR, Geneva, 2015.

[9] T. B. Brown et al., “Language models are few-shot learners,” in Proc. 34th Conf. Neural Inf. Process. Syst. (NeurIPS), 2020.

[10] OpenRouter, “OpenRouter unified interface for LLMs,” 2024. [Online]. Available: https://openrouter.ai/docs

[11] W. McKinney, “Data structures for statistical computing in python,” in Proc. 9th Python in Science Conf. (SciPy), 2010, pp. 51–56.

[12] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi, “The curious case of neural text degeneration,” in Proc. 8th Int. Conf. Learn. Representations (ICLR), 2020.

[13] OpenAI, “OpenAI Python Library,” 2024. [Online]. Available: https://github.com/openai/openai-python

[14] M. Imran, P. Mitra, and C. Castillo, “Twitter as a lifeline: Human-annotated twitter corpora for NLP in social services,” in Proc. 10th Int. Conf. Lang. Resources and Evaluation (LREC), 2016.

[15] L. D. R. E. De Silva, "Mini-Project-00: Operation Ditwah Crisis Intelligence Pipeline," 2025. [Online]. Available: https://github.com/damiengaming98/Mini-Project-00