AI-Powered Disaster Information Dashboard Using OpenAI APIs

# Sivaramakrishnan

Dept. of Computer Science and Engineering

Presidency University, Bangalore, India

Email: [sivaramakrishnan@presidencyuniversity.in](mailto:sivaramakrishnan@presidencyuniversity.in)

# Bhavana B A

Disha R

Dept. of Information Science and Engineering

Presidency University, Bangalore, India

Email: [disha.rr343@gmail.com](mailto:disha.rr343@gmail.com)

# Monika P

Dept. of Information Science Engineering

Presidency University, Bangalore, India

Email: bhavanagowda873@gmail.com

Dept. of Information Science Engineering

Presidency University, Bangalore, India

Email: [pmonikachowdary@gmail.com](mailto:pmonikachowdary@gmail.com)

***Abstract*—****This paper describes a real-time aggregation of disaster information that recognizes and visualizes current disaster incidents based on natural language queries. Dynamic data is fetched, mostly using OpenAI GPT-4, based on prompt-based queries and zero-shot learning. The backend, implemented using Node.js, generates prompts for GPT-4 that returns structured disaster information. A frontend based on React.js is used to display this information on interactive maps based on Leaflet.js and visualizations based on Chart.js. It locates specific disaster sites based on latitude and longitude and allows filtering based on type of disaster along with severity. Without any reliance on a conventional database, the design utilizes Natural Language Understanding (NLU) along with real-time inference to deliver dynamic, current results.**

**KeyWords: GPT-4, Zero-shot Learning, Prompt Engineering, Natural Language Understanding (NLU), Geospatial Mapping, Leaflet.js, Node.js, React.js**

1. Introduction

Real-time information of the disaster status is essential to be safe, make optimal choices, and respond quick enough in case of an emergency. The conventional mechanism of obtaining disaster information, however, involves searching different platforms or relying totally on the media, which will not be the most recent or location-specific information at all moments. There is a demand for a better mechanism that can gather real-time disaster data and provide the consumer with simple yet understandable information.  
In order to fill this gap, a real-time disaster information aggregator platform has been established. It compiles real-time disaster information based on natural language queries, and this is analyzed by advanced AI technology. This allows the user to determine current disasters, discover the kind of disasters, their intensity, and where exactly such accidents are situated based on the latitude and longitude co-ordinates. One can also narrow down the information based  
on both the nature of the disaster and its severity, and by visualizing data with the aid of interactive maps and charts to provide increased information clarity.  
The system utilizes OpenAI's GPT-4, that was chosen due to the advanced language processing that it can provide. GPT-4 has the capability of zero-shot learning, where it is able to process real time queries without needing to be retrained to adapt to varying sets of disaster information. This is adequately suited to a dynamic, real time system where disaster information continues to change. With the capability of generalization, GPT-4 is able to provide suitable insights based on general sources of information with a lack of the lag associated with typical machine learning models.  
The architecture of the system includes a frontend made with React.js to make the system user-friendly, and a backend made with Node.js that interacts with the GPT-4 to get the queries, process them, and obtain the disaster data. The backend requests well-ensured queries to the GPT-4, and in response, the data is returned in a real-time structured format. This data is visualized with Leaflet.js to be displayed over the map and visualizations with Chart.js and displayed to the user with the overall general view of the ongoing disasters. The system does not include a standard database, and hence, it fetches the data in real time based on requirement, leading to quick and scalable data fetching.  
With the real-time processing and zero-shot learning capability of GPT-4, the system can provide the latest, accurate information at any time without relying on the need to constantly update or retrain. This renders it an effective tool to stay informed and make instant decisions, especially when organizing travel or in a situation of emergency.

1. Related Work

The use of Artificial Intelligence (AI) and Natural Language Processing (NLP) has augmented real-time data collection and analysis and made informed decision-making more efficient. A number of studies have applied various methods like Named Entity Recognition (NER) and sentiment analysis to leverage geospatial mapping to facilitate better disaster monitoring and response.

Patel et al. [1] created an AI-based earthquake response system with real-time data extraction and deep learning-based analysis. It proved the necessity of mechanization of event classification, as is the case with Bidirectional Encoder Representations of Transformers (BERT) in disaster detection.

Kumbam and Vejre [2] introduced FloodLense, an AI-based flood detection framework that leverages ChatGPT for analyzing real-time flood reports. This study highlights the effectiveness of NLP techniques in identifying disaster-related information, supporting the use of models like BERT and Named Entity Recognition (NER) in disaster classification. Le [3] and Chanda applied BERT-based models for classifying disaster-related tweets, achieving high precision in filtering relevant information. Their research underscores the importance of deep learning in disaster detection, validating the application of BERT in our project for event classification and information extraction from social media. [4]

Ketmaneechairat and Maliyaem [5] investigated Conditional Random Fields (CRF) to process disaster-related text. CRF supports efficient entity extraction, and this aligns with the application of Stanford NER Tagger and SpaCy to extract disaster-related entities like locations and event types in our system.

Xu [6] discussed NLP and sentiment analysis progresses with a focus on the improvement of severity evaluation by deep learning-based models. This is consistent with applying VADER (Valence-Aware Dictionary and Sentiment Reasoner) in our work to evaluate sentiment in social media and measure disaster intensity.

Ariyabandu [7] researched Free and Open Source Software applications such as the Sahana Disaster Management System, highlighting the use of AI platforms in automating the workflows of disaster response. This facilitates the use of automated geospatial mapping and real-time visualization of disasters.

Rajabifard et al. [8] have explained how to build spatial data infrastructure to manage disasters, emphasizing the role of geolocation methods. In this system, the extracted locations are upgraded to be in terms of latitude-longitude format by the API of OpenAI to facilitate proper disaster visualization.

Wallace and De Balogh [9] indicated the part played by decision support systems in disaster management and illustrated that AI-based dashboards enhance response effectiveness. This aligns with integration of a React.js and D3.js dashboard in our project to monitor disasters in real time.

Bass et al. [10] laid emphasis upon software architecture principles that are needed to design a scalability.able AI-based disaster management framework. This makes sure that the AI system deployed in our project is efficient and flexible.

Bharosa [11] presented net-centric information orchestration to public safety networks to provide reliable and quality data processing.

1. METHODOLOGY

The Disaster Info Aggregation Dashboard is based upon a real-time data platform that is constructed upon an OpenAI GPT-4 framework. The platform fetches and processes data related to a disaster by applying Prompt-based Querying and LLM-based Zero-shot Learning. It provides the user with the latest disaster data, and there is no need for a standard database. Three symbiotic components make up the system architecture to provide real-time data, namely the frontend, the backend, and the OpenAI GPT-4 API.

The architecture works as follows

As illustrated in fig.1 The frontend, built with the React.js library, permits the user to ask questions related to disasters. When the user query is received, the frontend sends an HTTP GET request to the backend.

The back-end, created by using Node.js and Express.js, takes the query and creates a prompt that is sent to the OpenAI GPT-4 API to conduct the zero-shot learning processing of the query to provide the resultant response.

The information is passed to the backend by GPT-4 and is subsequently processed and returned to the frontend. The frontend renders the information through Leaflet.js to provide maps depicted in fig.5 and Chart.js to provide graphs.

The system makes use of real-time updates through WebSocket or long polling to ensure that information is kept fresh and displayed in accordance with what is current at every point in time.

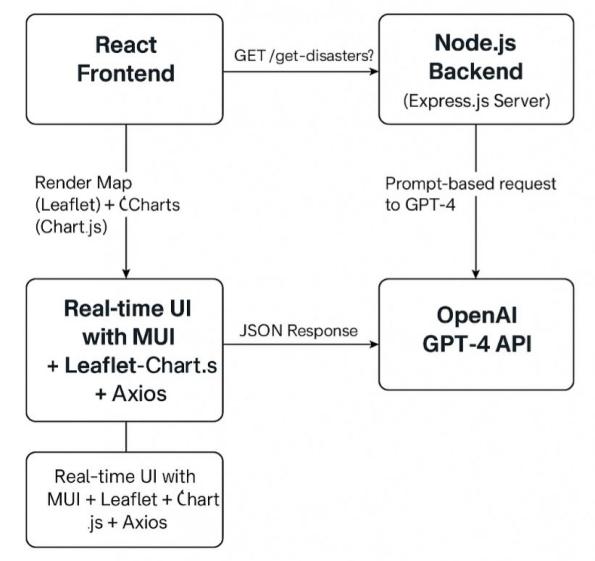
The Disaster Information Aggregation Dashboard collates disaster-related queries in real time through the use of OpenAI GPT-4, combined with sophisticated methods such as Prompt-based Querying and LLM-based Zero-shot Learning. This is not, however, done with a typical database. Instead, the Dashboard takes data directly from the latest version of GPT-4 when a query is posted, and therefore viewed data is always in real time.

When the frontend is requested, the request is submitted to the backend (Node.js and Express.js-based). The backend generates a prompt that will extract the most pertinent disaster information from GPT-4. The prompt has been designed by applying the technique of prompt engineering to ensure that the model produces useful and correct responses.

When the prompt is sent to GPT-4, the model applies the method of zero-shot learning to the query and hence produces its predictions without any such training with disaster-related data. In contrast to the necessity to directly get trained with disaster data, GPT-4 makes use of previously gained knowledge with Natural Language Understanding (NLU) to identify the meaning of the query and make appropriate inferences.

The response of GPT-4 is sent to the backend, where data is processed and returned to the frontend. Frontend data is rendered with Leaflet.js to visualise disaster location mappings and Chart.js to visually show disaster-related data. Frontend is designed to update data in real time by either WebSocket or long polling, ensuring that the user receives only the freshest disaster data.

Since data is not stored by the system in a standard database, with each query there is a new call to GPT-4, and thus the system is dynamic and scalable. Owing to there being the use of zero-shot inference, the system is flexible enough to match disaster-related queries, and thus the system can respond to any type of query without specialized datasets. In summary, the Disaster Information Aggregation Dashboard leverages Prompt-based Querying to LLM-based Zero-shot Learning using OpenAI GPT-4, and this is performing the queries in real time. Owing to the use of Natural Language Understanding (NLU) and prompting, and zero-shot inference, the platform is able to effectively process disaster data and at a very swift rate without a standard database. Inference and real-time fetching of data make the system highly efficient and responsive to requests.



*Fig. 1. System Architecture and Design*

1. RESULTS and DISCUSSION

The Disaster Info Aggregation Dashboard proved to be highly efficient in the processing of real-time disaster questions with the help of GPT-4 and advanced Natural Language Processing. One could identify ongoing disasters, organize the data in a comprehensible format, pinpoint their positions, and provide useful data such as severities and event distribution all of this without recourse to any typical database.

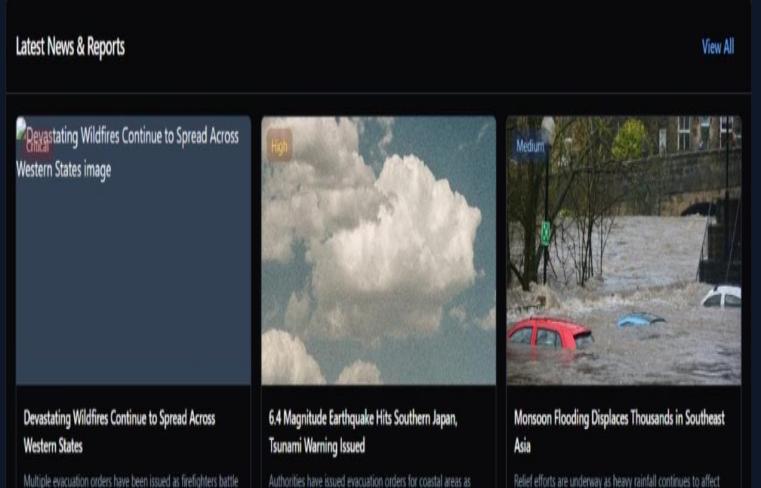
Two of the basic strategies at the heart of the system are Prompt-based Querying and Zero-shot Learning, both driven by the GPT-4 model. Both strategies feed off each other to understand the background of the query and generate accurate responses, regardless of how new the query is.

Prompt-Based Querying consists of giving well-constructed instructions (or prompts) to GPT-4 that cause it to respond with a specific type of information. For instance, the back end that is built with Node.js and Express.js gives such prompts such as:

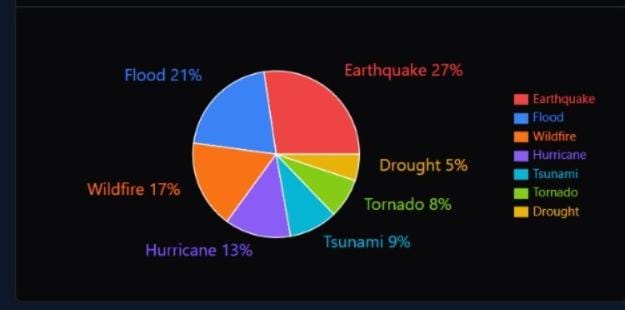
The list of current natural disasters in the world with the disaster type, location, date, severity, and the associated latitude and longitude is returned by GPT-4 in the exact format requested, and in the correct format and organization.

On the contrary, Zero-shot Learning permits the model to respond to questions that are not explicitly trained upon. As GPT-4 has been pre-trained with a very large amount of data, with this generalizability, it can respond to a wide range of different disaster-related queries without the need for further training. This enhances the power but versatility of the system.

GPT-4 is based on a transformer architecture that uses self-attention, positional encoding and dense layers to understand each word based on what is around it. It generates text sequentially, relying on previous words and context to forecast what will come next something that is essential to providing accurate, well-structured responses to disaster questions.

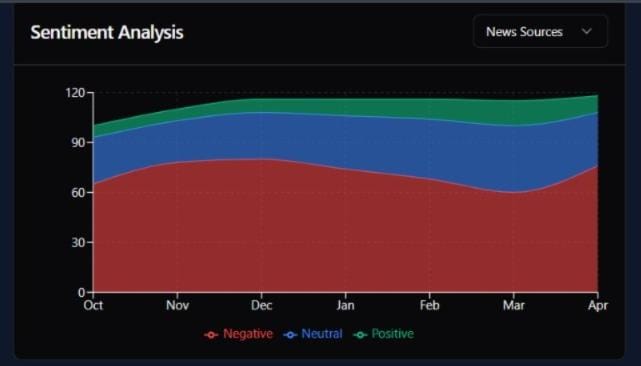


*Fig.2. Image depicting real time analysis through news alerts*

The model also carries out functions like place detection, disaster type, and levels of severity — all constituents of Natural Language Understanding (NLU) functionality. Even though we don't manually train the model to detect those characteristics, the model picks them up by itself because of its advanced language understanding 

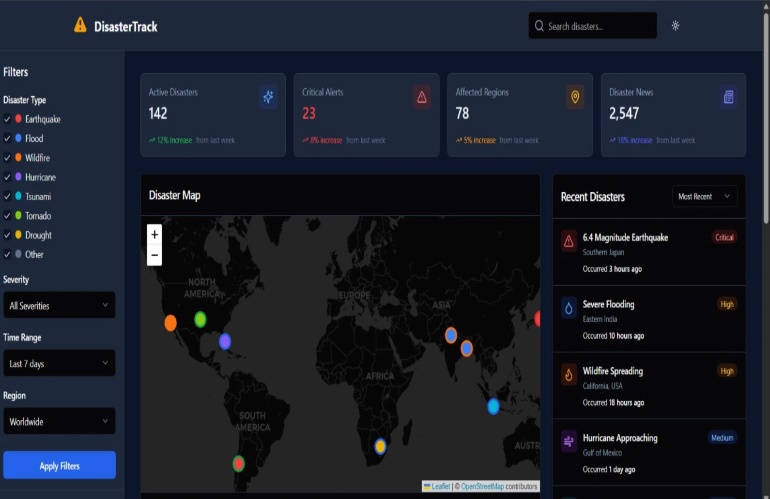
*Fig.3. Pie Chart Representing disaster distribution percentage across the world.*

All the data gathered with the help of GPT-4 is rendered via an intuitive interface that has been designed with Leaflet.js and React. Such disasters are visualized in the format of pins over the map of the world like in fig.5, with the help of the latitude and longitude coordinates that have been offered by the model. Derived visual overviews are also prepared with the help of Chart.js with the aid of bar and pie charts like in fig.3 to showcase the nature of the disaster, the frequency with which the disaster occurred, and the severity with which the disaster happened. As indicated in fig.4 sentiment analysis chart is offered too, reflecting the emotional tone over the reports of the disaster interpreted by GPT-4.



*Fig.4. Sentiment Analysis Flow*

No data is stored in any database, so each query fetches new, live results. This not only renders data in real-time, but reduces the privacy and security threat. We manage configurations via environment variables and use Axios and CORS to facilitate secure communication between the frontend and the backend.

In all, integrating Prompt Engineering, Zero-shot Learning, and Transformer-based Natural Language Understanding, the system is able to generate accurate, real-time disaster intelligence with lightweight, efficient, and cutting-edge architecture.

*Fig.5. Disaster spotted across the world on world map*

5 CONCLUSION

The Dashboard of Aggregation of Disaster Information employs cutting-edge technologies such as GPT-4 to fetch recent disaster data, providing an interactive and dynamic experience. Based on prompt-based asking and the idea of zero-shot learning, the system handles the queries asked by the user with respect to world disaster incidents effectively, projecting the same through an intuitive and clean interface. GPT-4 ensures that there is no special training or data sets, and therefore that remarkable level of flexibility and adaptability is displayed by the dashboard. The disaster information is displayed through the dashboard in dynamic charts and maps, all real time, but without persistent storage.

The solution thus excludes the use of a database in that a user query triggers a direct, real-time process to fetch data. This means the system produces insights that are current to the minute without the threat of data retention challenges. The architecture is minimal and secure, and the frontend and the back end interact seamlessly with technologies such as Axios and CORS.

Future development will include expanding disaster data feeds, increased support for many languages, adding capability to send alerts to storms, floods, landslides, and earthquakes in real time, and adding more sentiment analysis to provide more meaningful insight to user sentiment.

6 REFERENCES

[1] R. Kumbam and K. M. Vejre, “FloodLense: A Frame- work for ChatGPT-based Real-time Flood Detection,” arXiv, 2024.

[2] D. Le, “Disaster Tweets Classification using BERT- Based Language Model,” arXiv, 2022.

[3] Ketmaneechairat and M. Maliyaem, “NLP for Disaster Management Using CRFs,” J. Adv. Inf. Technol., 2020.

[4] Ariyabandu, “FOSS for Disaster Management: Sahana Case Study,” UN ESCAP, 2009.

[5] Xu, “Advancements in Deep Learning and NLP for Effective Disaster Sentiment Analysis: A Review,” Appl. Comput. Eng., 2024.

[6] K. Chanda, “Efficacy of BERT embeddings on predict- ing disaster from Twitter data,” arXiv, 2021.

[7] A.Wallace and F. De Balogh, “Decision Support Systems for Disaster Management,” Blackwell, 1985.

[8] Patel, P. Bhattacharjee, A. Reza, and P. Pradhan, “Earthquake Response Analysis with AI,” arXiv, 2025.

[9] Bass, P. Clements, and R. Kazman, “Software Archi- tecture in Practice,” 2nd ed., Pearson, 2003.

[10] Rajabifard et al., “Developing Spatial Data Infrastruc- ture for Disaster Management,” Proc. GEOMATICS’83, 2003