IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Anomoly Detection In Log Files Using Genai

[1]Mohammed Rayyan Faizan, [2]Nithessh Kumar, [3]Mohd Ehteshaam Khan, [4]Keshav Sharma, [5]Pallavi B

[1,2,3,4]Student 4th Year B.E. , [5]Associate Professor

[1,2,3,4,5]Dept. of Machine Learning,

[1,2,3,4,5]B.M.S College of Engineering, Bangalore, India

Abstract: This Log analysis is a critical process in IT operations, often requiring significant time and technical expertise. Traditional methods are manual, slow, and prone to errors during incident resolution. This project presents GenAI LogAnalyzer, an AI-powered system that automates log analysis using Large Language Models (LLMs). The tool leverages semantic search and natural language processing to identify anomalies, trace prompt flows, and assist in root cause analysis. It integrates with frameworks like LangChain and platforms such as ServiceNow to streamline incident handling. Logs are converted into structured insights using vector embeddings and LLM-based summarization. Featuring a user-friendly interface, the system enhances scalability, reduces operational overhead, and accelerates IT support workflows with intelligent, automated analysis.

Index Terms - Generative AI, Large Language Models (LLMs), Semantic Search, Vector Embeddings, Anomaly Detection, Root Cause Analysis, Automated Troubleshooting, LangChain Framework

I. INTRODUCTION

Modern IT systems generate extensive logs that are crucial for debugging, auditing, and optimizing workflows—especially in environments driven by Large Language Models (LLMs). Traditional log analysis methods are manual, time-intensive, and often insufficient for handling the complexity of AI-based pipelines. Anomaly Detection using GenAIis an AI-driven system purpose-built to analyze and interpret system-generated logs from advanced, multi-step workflows. It is specifically tailored for platforms using LangChain or custom LLM frameworks, where existing tools fall short. The system leverages natural language processing, semantic search, and embedding-based retrieval to trace prompt-response flows, detect anomalies, and extract actionable insights from unstructured logs. Key features include prompt-level tracing, LangChain log analysis, and token usage monitoring. This project aims to support IT operations with faster, consistent, and scalable diagnostics, ultimately improving system transparency, reliability, and performance.

II. LITERATURE REVIEW

This section reviews key advancements in the application of large language models (LLMs), prompt engineering, and deep learning to the domain of log analysis, anomaly detection, and automation in IT operations. These contributions collectively form the foundation of the proposed GenAI LogAnalyzer system, which integrates state-of-the-art language models via LangChain and OpenAI APIs to automate and enhance log interpretation.

A. Prompt Engineering and Pattern Guidance

[1] Chen et al. (2023) introduced a Prompt Pattern Catalog to standardize and improve prompt engineering with foundation models such as GPT-3. The study presents reusable templates that enhance prompt consistency and reduce hallucinations, providing a systematic approach to interacting with LLMs for structured tasks like log analysis. However, the study lacks integration with real-world observability pipelines and offers limited quantitative evaluation in IT-specific contexts.

B. Explainable Log Anomaly Detection

[2] Arora et al. (2023) developed a framework for explainable log anomaly detection by fine-tuning LLMs on structured log sequences. Their method employs masked log prediction and attention-based explanations, achieving robust detection performance across datasets. Despite its accuracy, the method is computationally intensive and shows reduced effectiveness on noisy or out-of-distribution logs.

C. Prompt-Based Time Series Forecasting

[3] Zhou et al. (2022) proposed PromptCast, a model that adapts LLMs for multivariate time series forecasting through prompt-based sequence modeling. While not focused exclusively on logs, the method's ability to capture temporal dependencies is applicable to structured log events. Limitations include the need for domain expertise in prompt tuning and dataset-specific performance variability.

D. LLM-Driven Log Anomaly Detection

[4] Zhang et al. (2022) presented LogGPT, an LLM-based system trained on log template sequences for few-shot anomaly detection. The model effectively captures long-term dependencies in logs and provides interpretable scoring. However, it is resource-intensive during inference and sensitive to the quality of input log templates.

E. Automated Log Remediation and Fix Suggestions

[5] Tufano et al. (2020) explored the use of Transformer-based models for automated code and log repair. Although originally intended for code, the approach demonstrates strong semantic understanding applicable to log correction and remediation workflows. The model, however, requires fine-tuning for ITSM contexts and lacks seamless deployment support.

F. Unified Log Parsing and Detection

[6] Han et al. (2023) proposed LlamaLog, a unified framework that uses instruction-tuned LLaMA-2 models for log parsing and anomaly detection. The system surpasses traditional parsers in robustness and adaptability across industries. Despite its effectiveness, the reliance on large LLMs poses challenges for scalability, multilingual support, and real-time performance in production environments.

III. Problem Statement

Traditional log analysis during IT incident resolution is manual, inefficient, and lacks intelligent insight. Support engineers are required to sift through vast volumes of unstructured logs, making the process time-consuming and highly prone to human error. Existing tools rely heavily on rule-based parsing and static filters, which are inadequate for handling complex, evolving system architectures and subtle failure patterns. Moreover, they are unable to provide root cause explanations or suggest remediation steps without significant manual effort. This leads to delayed incident resolution, increased downtime, and a higher operational burden on IT teams. With the increasing complexity of AI-driven workflows and the need for rapid, reliable diagnostics, there is a growing demand for automated, intelligent log analysis systems. The challenge lies in developing a scalable AI-based solution capable of interpreting logs from environments like LangChain, detecting anomalies, tracing prompt-response interactions, and generating actionable insights—while ensuring accuracy, adaptability, and efficiency in real-world IT operations.

IV. Scope and Objectives

A. Scope

The **GenAI – LogAnalyzer** project addresses the pressing need for intelligent and automated analysis of complex system logs within modern IT infrastructures. Traditional log analysis techniques are often manual, time-intensive, and lack the contextual intelligence required to effectively diagnose and resolve incidents—particularly in environments where AI systems are deeply integrated. This project aims to deliver a scalable, AI-powered platform that enhances incident detection, root cause analysis, and overall operational efficiency by harnessing the capabilities of Large Language Models (LLMs) and modern frameworks.

Designed for deployment in enterprise IT operations, the system functions as an intelligent incident support tool that delivers real-time, actionable insights. It is specifically optimized for LLM-centric workflows, such as those involving LangChain or custom AI pipelines, where tasks like prompt tracing, anomaly detection, and explainability are essential. The project scope spans several key areas. It begins with automated log data collection and preprocessing, enabling the system to ingest and structure logs from diverse sources—including

applications, infrastructure components, and AI workflows—while applying timestamp parsing, noise filtering, and prompt-response correlation to prepare data for analysis. The system then applies semantic analysis and AI inference techniques to detect anomalies and diagnose performance issues or errors, offering explanations and identifying root causes using context-aware language models. To further enhance interpretability, the project incorporates advanced prompt engineering strategies that are specifically crafted for extracting insights from LLMs, with a focus on debugging and understanding logs generated by generative AI systems.

Another major focus is seamless integration with existing IT ecosystems, ensuring compatibility with platforms such as ServiceNow, Datadog, and other monitoring or service management tools for real-time alerting and visualization. Transparency and explainability are also central to the system's design, with features like attention visualization, prompt-flow graphs, and trace reconstruction helping IT teams understand the rationale behind each decision. To support enterprise-scale deployment, the architecture is built to be modular, cloud-native, and optimized for performance using platforms like AWS Bedrock and SageMaker, thereby enabling rapid scaling, low-latency operation, and adaptability across diverse infrastructure environments.

B. Objectives

The GenAI – LogAnalyzer project is focused on transforming and automating IT incident management by leveraging the capabilities of Generative AI to analyze complex log data. The core aim is to develop an intelligent system that can parse and interpret vast volumes of unstructured log files with minimal manual effort, thereby streamlining traditional log analysis workflows. A key objective is to implement robust anomaly detection mechanisms capable of identifying deviations and error patterns across diverse systems, including applications, infrastructure, and AI pipelines. The platform also aspires to deliver accurate, AI-generated root cause analyses that not only explain anomalies but also suggest potential remediation steps, significantly accelerating the resolution process.

To ensure seamless operation within enterprise environments, the system is designed for real-time integration with existing IT monitoring and incident response frameworks, promoting continuous operational flow. Additionally, a strong emphasis is placed on explainability and interpretability, enabling users to understand and trust the reasoning behind AI outputs. This is crucial for adoption in production settings where decision accountability is essential. Performance and scalability are also central to the system's architecture, allowing it to handle high volumes of log data with low latency while maintaining cost-effectiveness. Finally, the design incorporates a modular structure that supports future extensibility, paving the way for enhancements such as predictive alerting, user behavior analytics, and integration with broader observability tools. Through these goals, the project aims to redefine how enterprises manage and respond to IT incidents using the transformative potential of generative AI.

V. Current System

The current landscape of log analysis in IT systems is dominated by a mix of manual techniques, scripting-based tools, and commercial platforms. While these approaches have proven useful in many settings, they struggle to keep pace with the growing demands of scalability, adaptability, and intelligent insight. Below is an overview of both traditional and emerging methods, highlighting their strengths and limitations.

A. Traditional Manual Systems

Many IT teams still rely on basic command-line utilities such as grep, less, and tail to manually inspect logs. While effective for straightforward tasks, these tools are inherently limited—they are time-consuming, prone to human error, and lack the ability to scale across multiple sources. Additionally, they are not equipped for real-time monitoring or complex event correlation, making them inadequate for dynamic or large-scale environments.

B. Scripting-Based Approaches (Regex/Parsing)

Developers often create custom scripts using regular expressions and parsing logic tailored to specific log formats to automate repetitive checks. These scripts provide some automation and flexibility but are rigid in nature, requiring significant programming expertise and ongoing maintenance. They frequently fail when faced with unstructured logs or unforeseen error patterns and lack the adaptability needed in modern systems.

C. Traditional Log Management Systems (e.g., ELK Stack, Splunk)

Platforms such as the ELK Stack and Splunk are widely used for centralized log collection and analysis. They enable real-time data aggregation, advanced querying, dashboards, and alerting mechanisms. However, these systems are resource-intensive to deploy and maintain, require familiarity with specialized query languages, and often depend on manual configuration for effective anomaly detection. Their ability to identify novel patterns without explicit rules is limited.

D. Proprietary AI-Based Systems

Some commercial tools now incorporate AI capabilities for pattern recognition and anomaly detection. While these systems can provide accurate results in known scenarios and offer advanced automation, they often function as closed-source solutions lacking transparency and explainability. High licensing costs and limited customization options also make them less accessible to smaller teams or organizations.

E. Limitations of Existing Systems

Across all categories, existing log analysis tools share key limitations: they often lack natural language interfaces, struggle with unstructured or unexpected log formats, require significant domain knowledge and configuration effort, and provide limited interpretability of their outputs. Additionally, the high infrastructure demands and costs associated with many solutions present substantial barriers to adoption, especially for smaller enterprises.

Summary

In summary, while traditional and commercial solutions offer varying degrees of functionality, they fall short of delivering intelligent, scalable, and user-friendly log analysis—gaps that the GenAI LogAnalyzer is specifically designed to address.

VI. Requirement Analysis

To ensure usability, intelligent functionality, and adaptability across diverse enterprise environments, the design of the GenAI LogAnalyzer is grounded in a comprehensive requirement analysis encompassing functional, non-functional, and technical aspects.

A. Functional Requirements

The system is designed to ingest and parse raw logs from a variety of sources, converting them into structured formats suitable for analysis. It incorporates advanced AI models—such as Transformers and Graph Neural Networks (GNNs)—to detect anomalies and classify events effectively. For root cause analysis, the system correlates log entries to identify potential failure points, complemented by severity and impact assessments to prioritize incidents. It also offers remediation suggestions based on historical resolution patterns and integrates with IT Service Management (ITSM) platforms like ServiceNow for automated ticket creation and tagging. A web-based interactive dashboard allows users to visualize log data, filter events, and monitor system health in real time.

B. Non-Functional Requirements

The platform is built to scale, capable of processing large volumes of enterprise-level log data across distributed systems. It ensures real-time performance with minimal latency in detection and response. Accuracy is prioritized through high precision and recall in identifying critical issues, while adaptability ensures effective performance across varied log formats, cloud platforms, and architectures. Security is addressed through adherence to enterprise-grade data protection and secure access protocols. The system is designed to be reliable and fault-tolerant, maintaining consistent performance under load. Explainability is embedded in the analysis pipeline, providing interpretable outputs such as visual explanations or attention heatmaps. Usability is a core focus, with an intuitive dashboard requiring minimal training for analysts and operators.

C. Technical Requirements

The underlying AI infrastructure leverages models based on Transformers and GNNs for learning from sequences and structured data. Model development is supported by frameworks like PyTorch or TensorFlow. The frontend is developed using modern, responsive tools such as React, Streamlit, or Dash, ensuring effective visualization and user interaction. Deployment is cloud-native, with compatibility across AWS, GCP, and Azure, and supports containerization through Docker. GPU acceleration is employed to enable rapid inference

and training. Finally, the system is designed to integrate smoothly with both external IT tools and internal enterprise systems through RESTful or API-based communication.

VII. Proposed System – Architecture

The proposed system is a modular, AI-driven platform for automated log analysis and natural language querying. Designed with extensibility and enterprise readiness in mind, it enables dynamic, scalable processing of large-scale logs across diverse IT infrastructures. Future improvements may include integration with incident management tools and user feedback loops for continual model refinement.

A. Proposed System

The system enables users to upload system and application logs, which are then parsed, vectorized, and semantically queried using natural language. Its architecture supports seamless integration with large language models (LLMs) through LangChain, delivering insightful and context-aware responses in a human-readable format. Key capabilities include semantic understanding of logs via embedding-based similarity search, support for intuitive natural language queries, and advanced contextual reasoning powered by LLMs such as the OpenAI GPT API. The system also ensures fast retrieval through the use of vector databases like FAISS or ChromaDB and offers an interactive graphical user interface for efficient input and output handling.

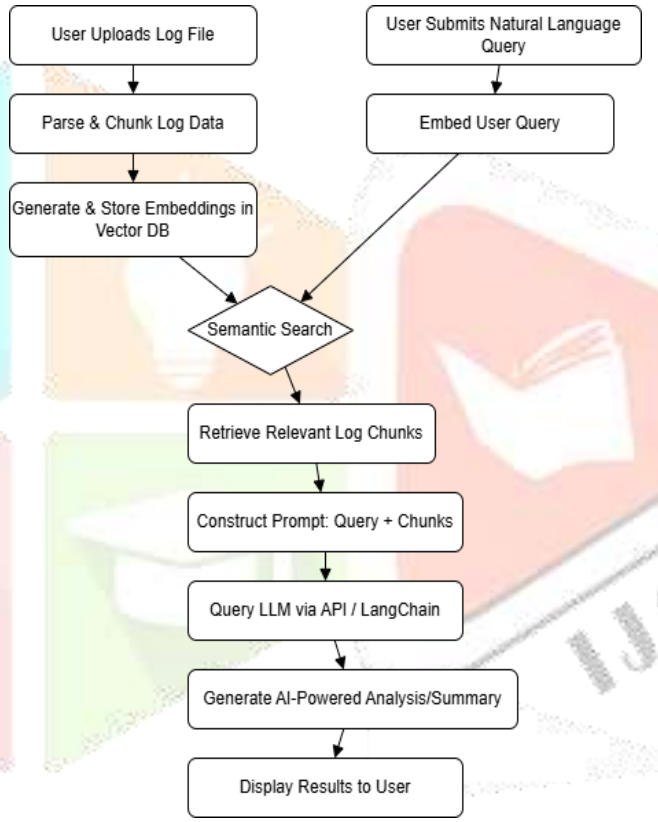

Figure 1 – Proposed System Design

B. High-Level Architecture

The architecture of the **GenAI LogAnalyzer** system is composed of six core components, each engineered for independent scalability and adaptability to suit a wide range of enterprise environments: log ingestion and chunking, vector embedding, semantic retrieval, prompt construction, LLM-based reasoning, and a user interface. These modules work in tandem to deliver an intelligent, end-to-end log analysis pipeline.

• The process begins with Log Ingestion and Preprocessing, where users can upload raw log files via a graphical interface or an API. These logs are automatically parsed and segmented into manageable chunks using intelligent splitting methods based on sentence boundaries or token limits. These chunks form the foundation for downstream semantic operations.

• Next, in the Embedding and Vector Storage phase, each log chunk is transformed into a vector representation using a transformer-based model (e.g., OpenAI Ada). These embeddings are stored in a high-performance vector database such as FAISS or ChromaDB, enabling fast and accurate similarity-based retrieval.

• The Natural Language Query Interface allows users to ask questions in plain English, such as "Why did the database crash at 2 AM?" These queries are embedded using the same model applied to the log chunks to maintain semantic consistency.

• In the Semantic Retrieval and Prompt Construction module, the system performs a semantic search to match the query embedding with the most contextually relevant log embeddings. These relevant chunks are combined with the user's original query to dynamically construct a prompt. This prompt is then passed to a Large Language Model—invoked through the OpenAI API via LangChain—to generate a coherent and context-aware response.

• The LLM-Powered Response Generation component enables the model to analyze the prompt and produce a concise, natural-language summary or diagnostic explanation of the issue. The response typically includes inferred root causes, key timestamps, and actionable recommendations, all derived from the context of the retrieved log data.

• Results are delivered through a user-friendly Interface and Result Display, which may be built using frameworks like Streamlit, Dash, or React. The interface displays the generated summary, relevant log excerpts, and provides options for report downloads. Its design ensures accessibility for both technical and non-technical users.

• Finally, a planned Feedback Loop and ITSM Integration will allow users to rate the relevance and accuracy of responses, feeding into a continuous learning system for performance enhancement through fine-tuning or active learning. Integration with IT Service Management (ITSM) platforms like ServiceNow is also envisioned to enable automated ticket generation based on LLM-generated outputs.

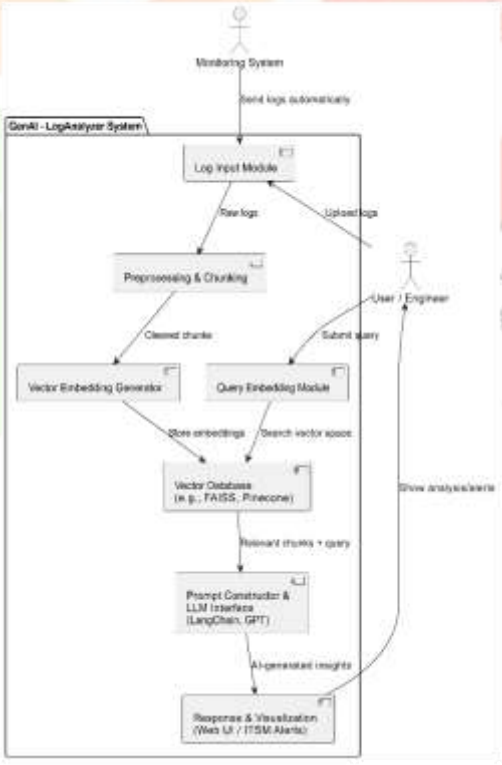

Figure 2 – High Level Design

This modular, intelligent architecture ensures that GenAI LogAnalyzer is not only powerful and flexible but also future-ready for ongoing enhancements and enterprise integration.

VIII. CONCLUSION

The increasing complexity of modern IT systems has led to a pressing need for intelligent and scalable tools capable of efficiently interpreting vast volumes of system and application logs. This project introduces GenAI LogAnalyzer, an advanced AI-powered platform designed to revolutionize log analysis using cutting-edge Large Language Models (LLMs) through LangChain and OpenAI APIs. The system automates critical tasks such as log chunking, semantic search, anomaly detection, and contextual summarization, removing the reliance on manual parsing or complex query languages. With a natural language interface and sophisticated embedding techniques, users can pose questions in plain English and receive accurate, insightful responses based on the underlying log data.

GenAI LogAnalyzer significantly enhances root cause analysis speed and reduces human effort, delivering substantial value across IT operations, DevOps, and cybersecurity domains. Its lightweight, modular architecture ensures broad compatibility with various log formats and vector databases, while also allowing for seamless scalability and integration. The platform is built with a strong focus on usability, featuring a simple yet effective graphical user interface, and places a high priority on security and data privacy—an essential consideration when dealing with sensitive infrastructure data.

Looking forward, the GenAI LogAnalyzer sets the stage for future innovations such as Retrieval-Augmented Generation (RAG) for real-time contextual insights, live stream log ingestion, integration with DevOps and SecOps pipelines for automated alerting and remediation, and feedback-driven learning loops to continuously improve performance. Ultimately, this solution showcases the transformative potential of generative AI in system diagnostics, offering a new paradigm for intelligent, explainable, and highly responsive log analysis in complex IT environments.

IX. REFERENCES

[1] LangChain Documentation. *LangChain*. (n.d.). *LangChain Docs*. Available at: https://docs.langchain.com

[2] OpenAI API Documentation. *OpenAI*. (n.d.). *API Reference*. Available at: https://platform.openai.com/docs

[3] ChromaDB Documentation. *Chroma*. (n.d.). *Chroma: AI-native vector database*. Available at: https://docs.trychroma.com

[4] Johnson, J., Douze, M., & Jégou, H. (2017). *Billion-scale similarity search with GPUs*. Facebook AI Research (FAISS). Available at: https://github.com/facebookresearch/faiss

[5] Streamlit for Frontend Interfaces. *Streamlit Inc.* (n.d.). *Streamlit Docs*. Available at: https://docs.streamlit.io

[6] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., … & Riedel, S. (2020). *Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks*. arXiv:2005.11401. Available at: https://arxiv.org/abs/2005.11401

[7] ServiceNow Integration Guides. *ServiceNow*. (n.d.). *Developer Documentation*. Available at: https://developer.servicenow.com

[8] Token Usage & Cost Tracking in OpenAI. *OpenAI*. (n.d.). *Pricing and Usage*. Available at: https://openai.com/pricing