Automated Root Cause Analysis in IT operations using machine learning and NLP

DISSERTATION

Submitted in partial fulfillment of the requirements of the

Degree : M.Tech in Data Science and Engineering

By

PORURI V S HARI HARA NANDAN
2022DC04433

Under the supervision of

ANIL KUMAR KANAPARTHI

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE
Pilani (Rajasthan) INDIA

December, 2024

Table of Contents

Abstract

Problem statement:

Understanding the root cause of problems in IT systems is crucial for effective troubleshooting and preventing future issues. However, traditional root cause analysis (RCA) methods often rely heavily on manual effort, leading to several challenges.

Firstly, manual RCA can be time-consuming, delaying the resolution of issues and impacting service availability. Secondly, human error is a significant risk in manual processes, potentially leading to inaccurate analysis and inefficient problem-solving.

Furthermore, managing and organizing RCA artifacts, such as logs, can be a complex and tedious task without robust systems in place. This disorganization can hinder collaboration and make it difficult for teams to access relevant information quickly.

Solution:

To address these challenges, automating the RCA process is essential. By implementing automated tools and systems, organizations can significantly reduce the time spent on RCA, improve accuracy, and enhance overall productivity. This allows teams to focus on proactive issue prevention rather than reactive problem-solving, ultimately leading to a more efficient and resilient IT environment

Broad Area of work

The broader area of work is to leverage ML and NLP to find anomalies in the logs and automate Root cause analysis process. This helps IT team for faster response and evaluation.

Objectives

The objectives of my project are to automate the Root Cause Analysis by ML, NLP and include below features and capabilities

Anomaly Detection and Event Correlation:

Machine learning models can quickly identify anomalies across vast datasets, linking them to specific events or patterns that contribute to system failures. By correlating events, they enable IT teams to pinpoint root causes faster, significantly reducing the time spent on manual log analysis and event mapping.

Knowledge Extraction:

Using advanced natural language processing (NLP), classification models extract meaningful insights from diverse data sources, including logs, incident reports, and historical records. This capability turns unstructured data into actionable insights, facilitating better decision-making during RCA.

Interactive Querying and Recommendations:

Classification models provide an interactive interface for IT teams to query the system and receive context-aware recommendations. For instance, they can suggest probable root causes, remediation steps, or preventive measures, empowering teams to resolve issues more effectively.

Automation of Repetitive RCA Tasks:

Time-intensive, repetitive tasks such as log analysis, report generation, and artifact classification are automated, freeing up human experts to focus on high-value activities like strategy and innovation.

Near Real-Time Processing:

By leveraging their computational power, classification models process and analyze logs and other data streams in near real-time, enabling IT teams to respond swiftly to incidents and mitigate their impact

Scope of work

The project is structured into four key phases to develop an interactive RCA system using classification model

Phase 1: Data Processing

Collect and preprocess system logs, performance metrics, and other relevant data from the IT environment.

Standardize, clean, and structure the data to ensure compatibility with AI models.

Phase 2: Anomaly Detection

Utilize AI-driven models to identify anomalies in the data, such as deviations from normal system behavior or performance thresholds.

Correlate anomalies with events, logs, and metrics to uncover potential causes of incidents.

Phase 3: Model performance tunning:

Try with different models and come up with the best ones.

Phase 4: Root Cause Analysis

Deploy transformer-based models to analyze the detected anomalies.

Provide clear, context-aware explanations of root causes by correlating multiple data points.

Leverage historical data and patterns to improve RCA accuracy and insights.

Phase 5: User Interface

Develop an interactive and user-friendly interface that enables IT teams to query the system directly.
Allow users to ask specific questions and receive actionable, AI-generated insights in real time.

Current status of the work

S. No	Task	Expected date of completion	Names of Deliverables	Current Status
1.	Abstract submission	09-DEC-2024	Abstract submission	Done
2.	Abstract Review from evaluator	5-Jan-2025	Received review comments from evaluator	Done
3.	Data Collection and exploration	17-JAN-2024	Collected open stack data from log hub	Done
4.	Data Preprocessing	22-JAN-2024	Used Drain parser to convert logs to structured format	Done
5.	ML Model Building	10-FEB-2024	Currently experimenting on different models	In progress
6.	ML Model evaluation	17-FEB-2024		
7.	UI Interface development	23-Feb-2024		
7.	Documentation/Unit testing	26-Feb-2024		
8.	Review with supervisor	28-Feb-2024		

List of Symbols & Abbreviations used

Libraries and packages used so far

Sno	Library	Uses
1	DRAIN	A log parser to convert unstructured data to structured data
2	BERT	Bi directional Encoder Representation from Transformers

Abbreviations for dataset used

We have used open stack, a cloud computing platform logs for our case study. Following are the services of Open stack that we could see in open stack logs.

Sno	Service	uses
1	Nova	Compute service
2	Neutron	Networking service
3	Cinder	Block storage
4	Keystone	Identity Service
5	Glance	Image service for VM images
6	Swift	Object Storage
7	Heat	Orchestration service

List of Figures:

Directions of work

Chapter :1 Objectives met so far

Phase 1: Data Processing

The following objectives are met as part of Data Processing.

Sno	Tasks	Outcome
1	Data Collection	Explored various datasets and extracted open stacks logs from loghub
2	Data Analysis	Has made EDA to understand the Data even more better and able to get more insights
3	Data Preprocessing	Used python regex to extract log message from unstructured log data
4	Data processing and parsing	Has utilized python Drain library to parse the unstructured logs and bring them to structured format

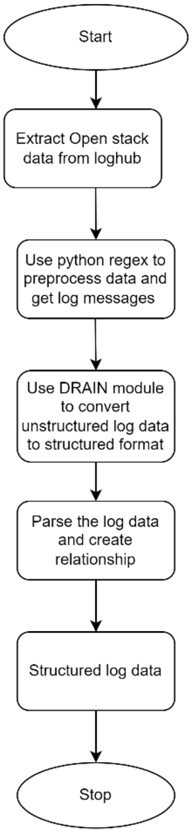

Start
Extract Open stack data from loghub
Use python regex to preprocess data and get log messages
Use DRAIN module to convert unstructured log data to structured format
Parse the log data and create relationship
Structured log data
Stop

Fig.1 Data processing flow chart

High level architecture

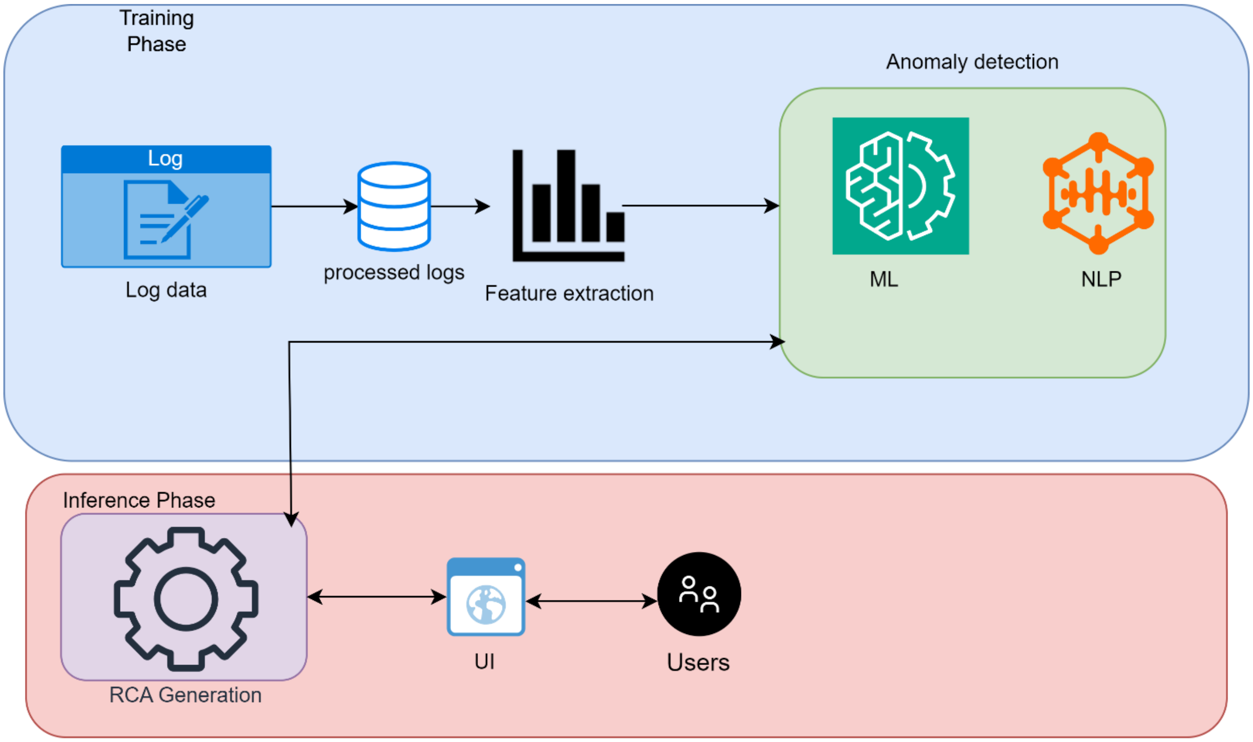


Fig 2: High level architecture

Exploratory Data Analysis:

The following EDA is performed so far

1. Input Logs

Normal Logs:

These are logs that represent standard, expected behaviors of the system.

Examples include routine application messages, system uptime notifications, or successful task completions.

Abnormal Logs:

Logs that indicate unusual or problematic behaviors in the system.

Examples include error messages, warnings, or anomalies like performance lags or failures.

2. Cleaning and Normalization

Purpose: Standardize the logs to prepare them for analysis.

Tasks Involved:

Remove Noise: Strip unnecessary characters or symbols (e.g., timestamps, redundant metadata).

Normalize Formats: Ensure consistency in the log formats (e.g., uniform date-time stamps, consistent naming conventions).

Once the data is processed, we are able to label the normal and anomalous data with 0 and 1.

Tokenization: Break down log messages into smaller units (tokens) for easier processing.

Logstructure

```
{
"log_file":"nova-compute.log.2017-05-14_21:56:26",
"log_timestamp":"2017-05-1421:55:05.347",
"pid":"2931",
"level":"INFO",
"logger":"nova.virt.libvirt.imagecache",
"req_id":"req-addc1839-2ed5-4778-b57e-5854eb7b8b09",
"message":"Activebasefiles:/var/lib/nova/instances/_base/a489c868f0c37da93b76227c9
1bb03908ac0e742"
}
```

3. Structure the logs

Purpose: Structure the logs to prepare them for analysis.

Tasks Involved:

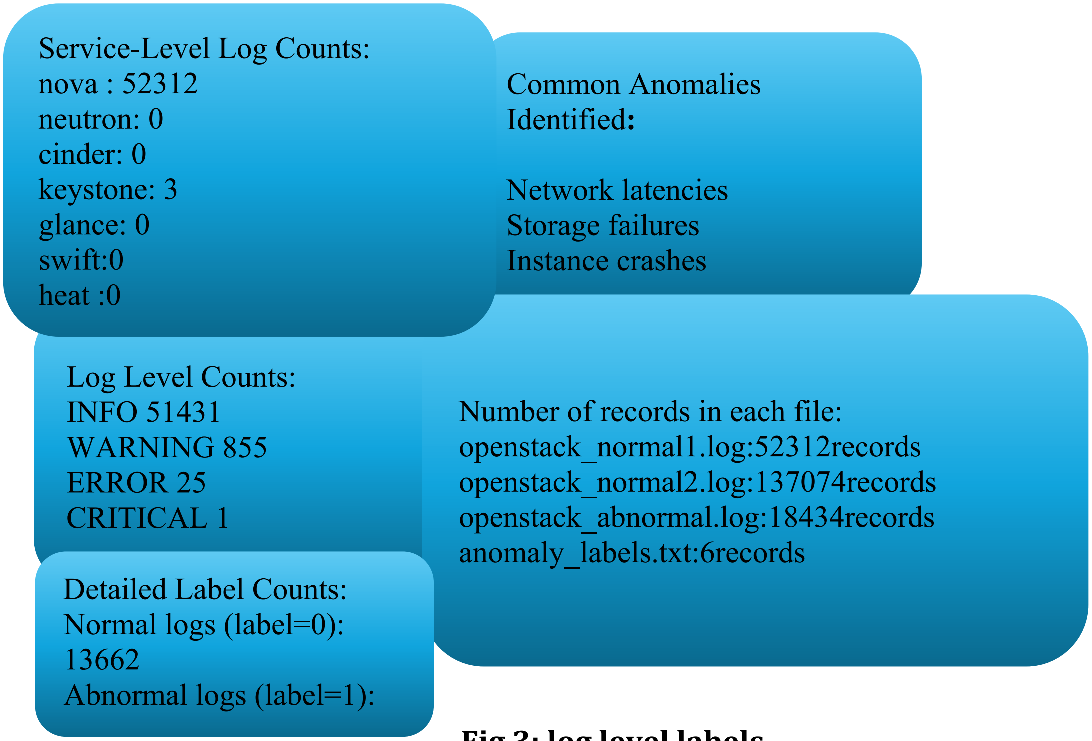
- ✓ Mine the template
- ✓ Create clusters using Drain 3 and attach the cluster ids

4. Attach the labels.

Purpose: Attach normal/Anomaly labels Structure the logs to prepare them for analysis.

Tasks Involved:

- ✓ Attach the labels



Service-Level Log Counts:
nova : 52312
neutron: 0
cinder: 0
keystone: 3
glance: 0
swift:0
heat :0

Common Anomalies Identified**:**

Network latencies
Storage failures
Instance crashes

Log Level Counts:
INFO 51431
WARNING 855
ERROR 25
CRITICAL 1

Number of records in each file:
openstack_normal1.log:52312records
openstack_normal2.log:137074records
openstack_abnormal.log:18434records
anomaly_labels.txt:6records

Detailed Label Counts:
Normal logs (label=0): 13662
Abnormal logs (label=1):

Fig 3: log level labels

Chapter :2 Current work in progress

Phase 2: Anomaly detection

Currently working on classical ML approaches like logistic regression, decision trees and logging the accuracy. Parallelly exploring Neural networks, and BERT model. Will compare with the best models as performance benchmark.

Chapter :3 Future targets

Phase 3: Model performance tuning

Will compare different models and come up with strategies for tunning the application.

Phase 4: Root Cause Analysis

Deploy transformer-based models to analyze the detected anomalies.

Provide clear, context-aware explanations of root causes by correlating multiple data points.

Leverage historical data and patterns to improve RCA accuracy and insights.

Phase 5: User Interface

Develop an interactive and user-friendly interface that enables IT teams to query the system directly.

Allow users to ask specific questions and receive actionable, AI-generated insights in real time.

Chapter :4 References

Reference 1 : M. Wang, L. Xu and L. Guo, "Anomaly detection of system logs based on natural language processing and deep learning", *2018 4th International Conference on Frontiers of Signal Processing (ICFSP)*, pp. 140-144, 2018.

Summary:

The paper by M. Wang, L. Xu, and L. Guo, titled "Anomaly detection of system logs based on natural language processing and deep learning," presents a method for detecting anomalies in system logs using natural language processing (NLP) and deep learning techniques. The authors propose using the doc2vec algorithm to construct sentence vectors from system logs. These vectors are then processed using various state-of-the-art classification algorithms to identify anomalies. The method was tested on logs from the Thunderbird supercomputer, demonstrating that the combination of doc2vec and machine learning algorithms effectively extracts semantic information and performs well in anomaly detection

Reference 2: https://medium.com/@lets.see.1016/how-drain3-works-parsing-unstructured-logs-into-structured-format-3458ce05b69a

Summary:

The article on Medium explains how Drain3, an online log template miner, works to parse unstructured logs into a structured format. Drain3 uses a fixed-depth parse tree to guide the log group search process, which helps avoid creating a deep and unbalanced tree. The tool continuously learns and extracts log templates from raw log entries in real-time. This process involves identifying patterns and creating templates that represent the structure of the logs, making it easier to analyze and monitor log data

Reference 3: IBM’s Log Analysis Tool.

Summary:

IBM’s Log Analysis Tool leverages AI to identify root causes of issues in IT operations by analyzing large volumes of system log data. The tool uses advanced algorithms to detect patterns and anomalies that may indicate potential problems, helping IT teams to quickly pinpoint the root causes of system failures or performance issues. However, a limitation of IBM's tool is its lack of interactivity, meaning users have limited options to engage dynamically with the analysis process or customize outputs based on their specific needs.