International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 08 Issue: 01 | January - 2024 SJIF Rating: 8.176 ISSN: 2582-3930

Smart Log Analyzerfor Anomaly Detection in Distributed Systems

Kartik Patil
Department of Information Technology
Pune Institute Of Computer Technology
Pune, India
kartikp3002@gmail.com

Dr. Shyam Deshmukh
Department of Information Technology
Pune Institute Of Computer Technology
Pune, India
sbdeshmukh@pict.edu

Rahul Sakharkar
Department of Information Technology
Pune Institute Of Computer Technology
Pune, India
rahulsakharkar2002@gmail.com

Saumitra Shinde
Department of Information Technology
Pune Institute Of Computer Technology
Pune, India
saumitra.k.shinde@gmail.com

Pratik Sonawane
Department of Information Technology
Pune Institute Of Computer Technology
Pune, India
prsonawane21@gmail.com

Mr. Pritam Fulsoundar
Senior Software Engineer
Veritas Technologies LLC
Pune, India
pritam.fulsoundar@veritas.com

***Abstract*—This research work outlines a novel approach to fortify the stability and security of distributed systems through the implementation of an AI-enabled smart log analyzer. The escalating proliferation of distributed systems has led to an unprecedented surge in the volume of generated log data, which holds crucial insights into system performance, security, and dependability. However, the substantial challenges associated with managing this data—such as its overwhelming volume, diverse nature, and real-time processing requirements—have posed significant hurdles. The proposed AI-enabled smart log analyzer, detailed in this report, harnesses the power of advanced machine learning and natural language processing techniques to address these challenges effectively. The methodology is structured into three fundamental phases, namely, Data Preprocessing, Anomaly Detection, and Clustering. The Data Preprocessing phase encompasses the collection, parsing, filtering, and feature extraction of log data. Anomaly Detection integrates machine learning models to discern various anomalies, encompassing irregular access patterns, log flooding, error messages, suspicious content, and outliers in resource requests. The Clustering phase categorizes log entries into meaningful groups based on attributes such as log level, component, event, error code, and resource usage, facilitating a comprehensive understanding of system behavior. This holistic approach holds the promise of significantly enhancing system stability and security.**

***Index Terms*—Anomaly Detection, Clustering, Data Preprocessing, Distributed Systems, Log analyzer, System security, System stability**

I. INTRODUCTION

Assuring the dependability, stability, and security of complex infrastructures has become a top priority in the world of modern distributed systems. For system administrators and operators, the exponential development in the volume and complexity of log data produced by these systems has presented serious difficulties. Innovative methods that can efficiently handle, analyse, and extract valuable insights from the enormous log data pool are required to address these difficulties. In order to do this, this research aims to propose a thorough framework for log clustering and anomaly detection in the context of distributed systems.

To enable effective and intelligent log analysis, the suggested framework makes use of AI technologies, such as machine learning models and natural language processing strategies. The system seeks to give administrators and system operators a strong tool for recognising unusual patterns, potential threats, and important system events within the OpenStack infrastructure by leveraging a rigorous methodology consisting of data pretreatment, anomaly detection, and clustering. The framework works to simplify the analysis of log data by incorporating advanced data processing algorithms and intelligent categorization processes. This enables quick and efficient decision-making for maintaining the stability and security of the distributed system.

Furthermore, the effective implementation of an AI-enabled smart log analyzer has the potential to revolutionise the management and monitoring of distributed system based settings in the modern cloud computing and distributed systems landscape. This research initiative represents a significant step forward in the quest for creating a reliable and resilient framework that can efficiently handle the intricate log data generated by distributed systems. This framework intends to improve the capacities of administrators, system operators, and security analysts in thoroughly monitoring and safeguarding the system infrastructure by solving the enduring difficulties connected with log data handling and analysis.

II. LITERATURE REVIEW

Numerous earlier studies using log data to investigate different machine learning techniques have been published. The complexity and volume of log data produced by distributed systems are constantly expanding, which has sparked an increased interest in creating efficient methodologies and tools for log analytics and anomaly identification. Every interaction, every service invocation, and every transaction within these systems produces an event captured in log files. These logs are a goldmine of information, containing valuable insights into system behavior, performance, and potential security threats. However, they also pose a significant challenge. The scale and diversity of log data make it impossible for human operators to manually review logs for anomalies effectively.

In a notable example from Microsoft[1], log clustering is employed for anomaly detection. This technique utilizes parsed and labeled log sequences with distinct weights and applies clustering algorithms to extract feature vectors. The resulting clusters are represented by a single value. When applied to new data, vectors are compared to cluster representatives. If a vector does not align with existing clusters, a new one is formed and manually labeled. Conversely, a match with a known cluster allows for automatic classification. This work introduced the innovative use of inverse document frequency (IDF) weighting for input vectors.

The effectiveness of several methods for log-based anomaly identification was examined in a study by S. He et al.[2] The authors also looked at unsupervised techniques like Principal Component Analysis (PCA) and invariants mining in addition to log clustering. Through dimensionality reduction, PCA identifies anomalous vectors depending on how far they deviate from main component projections, however it produced unsatisfactory results.

The research work[3] by Akanle et al. presents a significant contribution to the field of anomaly detection in cloud computing environments. The study leverages log data and machine learning techniques to develop an anomaly detection model tailored for OpenStack private cloud infrastructure. While the paper successfully demonstrates the effectiveness of Support Vector Machine (SVM) classifiers in identifying anomalies within system logs, there exist several areas where further research can enhance the model's applicability. These include the scalability and generalizability of the model to different cloud infrastructures, the real-time detection capabilities, integration with existing cloud security tools, and the interpretability of anomaly alerts. Additionally, exploring how to handle imbalanced datasets and categorize anomalies can contribute to a more comprehensive anomaly detection framework in cloud environments.

Log key anomaly detection and parameter value anomaly detection are the two main parts of the cutting-edge technique, DeepLog, developed by Min Du et al.[4]. The underlying presumptions of the first element are shared with Cloudseer. However, the authors suggest using a Long Short-Term Memory network (LSTM) rather than directly building a sophisticated automata, which can be difficult given the various interleaving possibilities and quickly changing cloud infrastructure source code. This LSTM builds a probability distribution for forecasting the upcoming log messages after learning the workflow by taking into account a window of previous log messages. Time series-based anomaly detection uses dynamic parameters that were collected from the logs during event template parsing to detect parameter value anomalies.

A. Das et al.[5] presented a fresh viewpoint in the constantly changing field of anomaly detection by utilising recurrent neural networks (RNNs). Within the sophisticated structure of supercomputing systems, their programme, Desh, focuses on the challenging task of node failure prediction. Desh adds a useful tool to the toolbox of anomaly detection approaches by utilising the temporal dependencies in log data.Additionally, X. Zhang et. al.[6] uses a novel strategy called LogRobust to address the widespread problem of log instability. It uses a sophisticated deep learning architecture called Bi-directional Long Short-Term Memory (Bi-LSTM). This approach excels in managing unstable log events and sequences, which is a crucial requirement in the area of practical log data analysis. The experimental results highlight LogRobust's effectiveness in reducing log instability, signalling a significant advancement in the field of log anomaly detection.

Alexander Emmerich et. al.[7] in 2018, looked at spotting oddities in logs from cloud installations. They used two methods: one based on word patterns and the other called DeepLog. What they found was that DeepLog worked really well in identifying these odd things. However, it's essential to realize that the logs from cloud installations are quite different from the OpenStack logs we're dealing with in this thesis. This highlights the need to tailor our approach to detect anomalies in distributed system logs since they have their own unique characteristics.

Previous research in the field of log analytics and anomaly identification has laid a robust foundation, focusing on various machine learning and deep learning techniques to detect anomalies within log data. However, several critical gaps persist, stemming primarily from the challenges posed by the scale and diversity of log data in complex distributed systems. These gaps include limitations in real-time detection capabilities, scalability to different cloud infrastructures, interpretability of anomaly alerts, and efficient handling of imbalanced datasets. Furthermore, the existing methodologies often lack the ability to effectively handle log instability and categorize anomalies, hindering their comprehensive applicability in practical scenarios. To bridge these gaps, the proposed project, the AI-enabled smart log analyzer, aims to integrate advanced machine learning and natural language processing techniques, enabling efficient log analysis, anomaly detection, and real-time monitoring. By leveraging innovative methods such as log clustering, time series-based anomaly detection, and recurrent neural networks, the project strives to enhance the accuracy and reliability of anomaly detection within log data, especially in the context of distributed system infrastructures. Furthermore, the project endeavors to address the

critical need for interpretability and adaptability in anomaly detection models, thereby providing a comprehensive and scalable solution for ensuring the stability and security of distributed systems.

III. PROPOSED METHODOLOGY

In this part, we present our suggested methodology for improving the stability and security of distributed systems by identifying anomalies in their logs. A large amount of log data is produced by a distributed system and is essential for tracking its performance. Our methodology seeks to improve the dependability and security of systems by efficiently finding and correcting abnormalities, ensuring seamless operations and preventive maintenance. The first pillar of our strategy is anomaly detection, which establishes the groundwork for early detection and action in the case of unexpected log entries. The organization and categorization of log entries are then made possible by clustering, thus increasing the overall efficiency of the anomaly detection process. In this methodology, we utilize various machine learning models to ensure accurate anomaly detection, and log clustering and enhance system stability.

A. DATA PREPROCESSING

A key part of our study is log processing, which aims to improve anomaly identification in the system logs to increase system security and stability. The first step in the process is log collecting, where we collect unstructured log data from various apps, servers, and network devices within the OpenStack system. Log parsing, a crucial process that includes dividing these unstructured logs into structured log entries, is next applied to this raw log data. Time stamps, log levels, and log messages are among the key components that are extracted using parsing rules.

After processing the logs, structured data filtering is used to keep only the necessary details and toss out the extraneous ones. This phase frequently entails filtering particular log data in accordance with the specific needs of our anomaly detection system. The process of turning the filtered log data into analytically useful numerical or categorical features is known as feature extraction. Techniques like Time Frequency-Inverse Document Frequency (TF-IDF), which are adapted from the field of text mining to build feature vectors, are used to make this transformation.

B. ANOMALY DETECTION

Anomaly detection within system logs is a pivotal component of this research aimed at improving system stability and security. The complex and extensive nature of log data generated in a distributed system environment demands effective techniques. This section explores various facets of anomaly detection, machine learning models, and their role in addressing distinct anomalies. These anomalies include unusual access patterns, log flooding, error messages, suspicious message content, and outliers in resource requests.

We harness the power of machine learning models like Random Forest, Isolation Forest, and Support Vector Machines to identify unusual access patterns such as a sudden increase in the number of attempts from a specific IP address within a short time frame. These models, trained on labeled log data, differentiate between normal and anomalous access patterns, quickly flagging unauthorized or suspicious activity. Additionally, preventing log flooding is crucial, and to do this, we use log throttling and rule-based filtering. Log throttling regulates the rate at which log entries are generated, and rule-based filtering removes pointless entries prior to anomaly identification.

Employing Natural Language Processing techniques, we categorize error messages into distinct classes using Word2Vec and other deep learning models. These NLP techniques allow the semantic analysis of error messages, aiding the early detection of anomalies. To scrutinize suspicious message content, we deploy NLP methods like sentiment analysis. These models identify patterns in log messages that could signify potential threats or security breaches. In the realm of resource request anomalies, the Isolation Forest algorithm identifies outliers efficiently, segregating normally from anomalous resource utilization patterns.

This systematic implementation of anomaly detection techniques ensures the prompt detection and mitigation of irregularities within the system logs. Timely responses to unusual access patterns, log flooding, error messages, suspicious content, and resource request outliers contribute to enhancing system stability and security. These methods are crucial in reducing system downtime, preventing security breaches, and optimizing resource allocation. By incorporating a diverse range of techniques, including machine learning and NLP models, our anomaly detection system adapts to dynamic log data within the distributed environment, maintaining high-performance standards.

C. CLUSTERING

In the context of contemporary computing settings, particularly in complex systems, proper administration and analysis of log data have become crucial. Log data, which is frequently vast and appears disorganised, hides important details about system operations, security risks, and performance irregularities. In distributed systems, log data can come from a wide variety of sources, such as various system processes and components. Understanding and gaining useful insights from such unstructured and diverse data might be likened to solving a complex jigsaw. Log clustering, which enables the collection of log entries with similar characteristics, acts as a crucial key to unlock these insights. It orchestrates the transformation of a jumbled array of log entries into ordered, cohesive sets, making it easier to understand system activity.

Log Level Classification: This pivotal classification scheme caters to the severity and importance of log entries. It enhances log clustering by categorizing logs into classes such as INFO, WARNING, ERROR, and CRITICAL. Here, the potential employment of natural language processing techniques plays a crucial role. NLP models may help gauge the sentiment

Fig. 1. Data Preprocessing of Distributed System Logs

and context of log messages, distinguishing between routine operational messages (INFO) and critical alerts (ERROR).

Component-Based Classification: In the OpenStack universe, log data emerges from a multitude of components. The art of classifying logs based on their originating component, such as Compute, Networking, Identity, or Storage, is invaluable. For instance, this classification enables the grouping of logs that pertain to the Nova Compute component under the ”Compute” class. Machine learning algorithms, including clustering and dimensionality reduction, can effectively differentiate between the components, enriching the log clustering process with structured knowledge.

Event-Based Classification: In our endeavor to classify logs based on specific events, we deploy a cocktail of ML models such as Recurrent Neural Networks (RNN) and Long Short-Term Memory networks (LSTM). These models, adept at sequence-based analysis, enable us to categorize logs according to events like VM Start, VM Stop, Authentication Failure, and Resource Allocation. The meticulous training and subsequent classification process formulates dynamic event clusters, where unexpected deviations can be swiftly identified. Error Code Classification: As logs are classified based on error codes and messages, Decision Trees, Random Forests, and Naive Bayes classification models take the reins. When a ”HTTP 404” class emerges, it signifies logs indicating page-not-found errors. The classification is empowered by ML models’ ability to discern subtle variations in error patterns, offering the potential for automated issue resolution and enhanced system stability.

Resource-Related Classification: In the realm of resource-based clustering, ML models such as Support Vector Machines (SVM) and K-Means Clustering provide insight into resource utilization. For example, the ”CPU Usage” class is nurtured by SVM, illuminating logs related to CPU performance. K-Means clustering dissects resource trends, enabling resource optimization, intelligent capacity planning, and performance enhancement.

Clustering log data within OpenStack logs provides a multifaceted advantage, with the foremost being a profound understanding of system behavior. Through clustering, we can uncover hidden patterns and group logs into meaningful categories, thereby enabling a holistic view of system activities. This enhanced comprehension contributes to efficient anomaly detection, proactive issue resolution, and optimized resource allocation. Moreover, clustering bolsters log analysis scalability, as patterns found in a single log instance can be extrapolated across the entire log ecosystem. This not only streamlines log management but empowers predictive maintenance and resource optimization. In essence, clustering acts as the cornerstone for effective log data analysis, bridging the gap

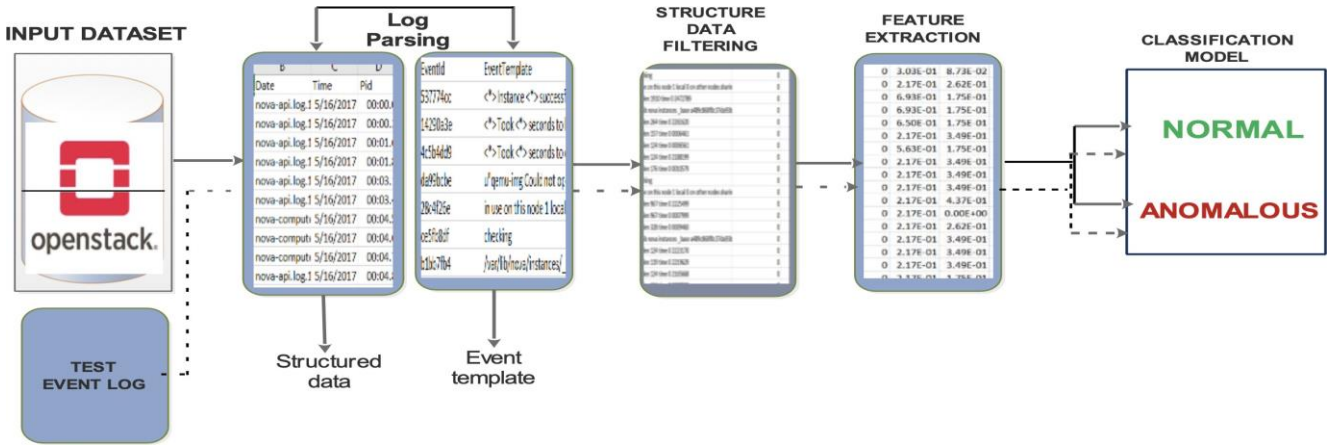

Fig. 2. Anomaly Detection model for Log Data Analysis

between unstructured logs and actionable insights, ultimately fortifying system stability and security within OpenStack.

IV. SYSTEM ARCHITECTURE

The implementation of the ELK (Elasticsearch, Logstash, and Kibana) stack, which acts as a potent and adaptable toolset for processing large-scale log data, is the central component of the project's system design. Elasticsearch, a distributed, RESTful search and analytics engine that can store, analyse, and analyse a massive quantity of data in real-time, sits at the centre of the architecture. The logs collected from multiple sources are conveniently stored in this data repository, ensuring easy access and retrieval for additional processing. Logstash serves as the pipeline for data processing, ingesting, converting, and moving log data from various sources to the Elasticsearch index. To facilitate the seamless flow of log data, Logstash employs a range of input, filter, and output plugins, allowing it to process various log formats and structures, ensuring consistency and coherence across the dataset.

In this research, the accuracy and effectiveness of log data analysis are improved by combining modern machine learning (ML) and natural language processing (NLP) approaches. To effectively identify anomalies and group log entries into discrete clusters, the system specifically makes use of the strength of supervised and unsupervised learning techniques, including Support Vector Machines (SVM), Random Forest, and K-means clustering. These machine learning algorithms can categorise log data according to predetermined patterns, structures, and qualities because they were trained on labelled datasets. To further extract useful information from the unstructured log data, natural language processing techniques including text tokenization, sentiment analysis, and named entity identification are used.

The system can perceive and interpret the contextual information contained inside the log entries by utilising these NLP approaches, enabling a greater knowledge of the underlying system behaviours and any abnormalities. The system's ability to recognise intricate patterns, correlations, and abnormalities within the log data is crucially aided by the integration of these ML and NLP techniques, which also improves the process' overall accuracy and efficiency.

Complementing the ELK stack, the system architecture integrates advanced machine learning frameworks, including TensorFlow and Scikit-learn, to implement sophisticated anomaly detection and clustering algorithms. These frameworks facilitate the development and deployment of customized machine learning models tailored to the unique requirements of log analysis in complex distributed systems. By leveraging the capabilities of these frameworks, the system can efficiently process and analyze the log data, identifying patterns, trends, and potential anomalies that might otherwise go unnoticed.

The architecture further incorporates Kibana, an intuitive and interactive data visualization platform that works in conjunction with Elasticsearch, enabling users to explore, analyze, and understand the log data effectively through customizable dashboards, charts, and graphs. This integration with Kibana empowers users to gain actionable insights and meaningful visualizations, contributing to a comprehensive understanding of the log data and enhancing decision-making capabilities. The system incorporates a user-friendly and intuitive graphical user interface (UI) to facilitate seamless interaction and navigation for system administrators, operators, and analysts. The UI is designed with a streamlined layout and visual representations of complex log data, enabling users to efficiently monitor, analyze, and manage the system's operational health and security status. Through interactive dashboards, customizable visualizations, and comprehensive reporting features, the UI provides users with real-time insights into log anomalies, cluster distributions, and system performance metrics. The intuitive design and user-centric features of the UI aim to enhance user experience and foster informed decision-making, ultimately contributing to the overall effectiveness and reliability of the log analysis system.

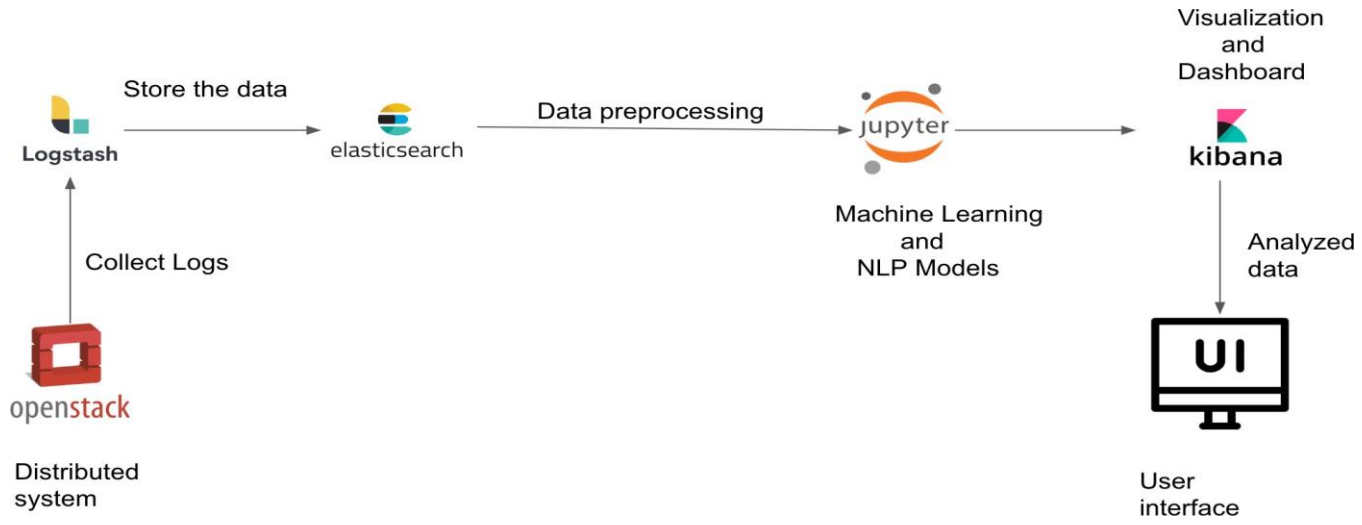

Fig. 3. System Architecture for the Smart Log Analyzer

V. CONCLUSION AND FUTURE SCOPE

In conclusion, this project has demonstrated the critical role that advanced machine learning techniques, natural language processing algorithms and data analysis play in enhancing the security and reliability of IT infrastructures, particularly within the context of distributed system environments. By implementing sophisticated anomaly detection algorithms and leveraging the power of natural language processing and machine learning, we have successfully identified and mitigated various anomalies and security threats within the system logs. These capabilities have bolstered the security posture of the system, enabling administrators to proactively identify and respond to potential security breaches and operational irregularities. The project's contribution lies in its ability to provide a proactive approach to cybersecurity, enabling administrators and system operators to detect irregularities, unauthorized access attempts, and suspicious behaviors before they escalate into significant security breaches.

Furthermore, the integration of advanced log clustering techniques has facilitated the efficient categorization and analysis of system logs, enabling users to gain deeper insights into system performance trends and operational health. The project's real-time alerting mechanism has enhanced the system's responsiveness to critical events and potential security threats, enabling users to take immediate actions to mitigate risks and ensure the system's integrity. Additionally, the interactive visualization dashboard has provided users with a comprehensive and intuitive interface for monitoring system activities, analyzing log data, and gaining actionable insights into the system's performance and security status. Overall, this project serves as a testament to the efficacy of AI-driven log analysis in enhancing the security posture and operational robustness of complex IT environments.

Looking ahead, the project opens up several avenues for future research and development. One potential direction is the integration of advanced deep learning models and neural networks to further improve the accuracy and precision of anomaly detection within OpenStack environments. By leveraging the capabilities of deep learning, we can enhance the system's ability to detect complex and nuanced anomalies, thereby fortifying the system's defenses against sophisticated cyber threats. Additionally, the project's scope can be expanded to encompass a broader range of anomaly detection and security monitoring techniques, including the integration of behavior-based anomaly detection and predictive analytics. This expansion would enable the system to proactively identify emerging security threats and potential vulnerabilities, thereby strengthening the overall resilience of the OpenStack infrastructure. Furthermore, the project can be extended to include comprehensive security auditing and compliance monitoring capabilities, ensuring that the OpenStack environment adheres to industry-specific security standards and regulatory requirements. This holistic approach to cybersecurity will foster a robust and secure IT ecosystem that can withstand the evolving challenges posed by modern cyber threats.

REFERENCES

[1] Q. Lin, H. Zhang, J. Lou, Y. Zhang, and X. Chen. "Log clustering based problem identification for online service systems". In: 2016 IEEE/ACM 38th International Conference on Software Engineering Companion (ICSE-C). 2016, pp. 102–111.

[2] S. He, J. Zhu, P. He, and M. R. Lyu. "Experience report: system log analysis for anomaly detection". In: 2016 IEEE 27th International Symposium on Software Reliability Engineering (ISSRE). 2016, pp. 207–218. DOI: 10.1109/ISSRE.2016.21.

[3] M. Du, F. Li, G. Zheng, and V. Srikumar. "Deeplog: anomaly detection and diagnosis from system logs through deep learning". In: CCS '17. Association for Computing Machinery, Dallas, Texas, USA, 2017, pp. 1285–1298. ISBN: 9781450349468. DOI: 10.1145/3133956.3134015. URL: https://doi.org/10.1145/3133956.3134015.

[4] Du M, Li F. Spell: Streaming parsing of system event logs. In: IEEE. ; 2016: 859–864.

[5] A. Das, F. Mueller, and C. Siegel, "Desh: Deep Learning for System Health Prediction of Lead Times to Failure in HPC", 40–51, 2018.

[6] X. Zhang, Z. Li, J. Chen, X. He, R. Yao, J.-G. Lou, "Cheng, Q. (2019). Robust log-based anomaly detection on unstable log data", 807–817. https://doi.org/10.1145/3338906.3338931.
[7] A. Emmerich. Automated Anomaly Detection in Software-Defined Telco Cloud Platforms. University of Applied Sciences Aachen Campus Jülich, 2018.
[8] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A Survey." ACM Computing Surveys, 41(3), 1–58. (2009). https://doi.org/10.1145/1541880.1541882.
[9] D. Grzonka, "The analysis of OpenStack cloud computing platform: Features and performance," Journal of Telecommunications and Information Technology, 2015.
[10] J. Zhu, S. He, J. Liu, P. He, Q. Xie, Z. Zheng, and M. R. Lyu, "Tools and Benchmarks for Automated Log Parsing" (pp. 121–130), 2019. https://doi.org/10.1109/icse-seip.2019.00021.
[11] T. Tokunaga, and I. Makoto, "Text categorization based on weighted inverse document frequency." In Special Interest Groups and Information Process Society of Japan (SIG-IPSJ), 1994.
[12] S. Beschorner. Automated Hyperparameter Tuning of Language Based Anomaly Detection for Log Files. University of Applied Sciences Aachen Campus Jülich, 2020.
[13] S. Alla and S. K. Adari. Beginning Anomaly Detection Using Python-Based Deep Learning With Keras and PyTorch. Apress, 2019, pp. 19–20.
[14] Farzad A, Gulliver TA. Unsupervised log message anomaly detection. ICT Express 2020; 6(3): 229–237.
[15] Makanju AA, Zincir-Heywood AN, Milios EE. Clustering event logs using iterative partitioning. In: ; 2009: 1255–1264.
[16] Vaarandi R. A data clustering algorithm for mining patterns from event logs. In: Ieee. ; 2003: 119–126.
[17] Tang L, Li T, Perng CS. LogSig: Generating system events from raw textual logs. In: ; 2011: 785–794.
[18] Fu Q, Lou JG, Wang Y, Li J. Execution anomaly detection in distributed systems through unstructured log analysis. In: IEEE. ; 2009: 149–158.
[19] S. Thakur, E. Adetiba, O.O. Olugbara, and R. Millham, "Experimentation using short-term spectral features for secure mobile internet voting authentication," Mathematical Problems in Engineering, 2015.
[20] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. "Lof: identifying density-based local outliers". ACM SIGMOD Record 29:2 (2000), pp. 93– 104. DOI: 10.1145/335191.335388. URL: https://doi.org/10.1145/335191.335388.