Received October 30, 2021, accepted November 7, 2021, date of publication November 15, 2021, date of current version December 2, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3128283

IT Infrastructure Anomaly Detection and Failure Handling: A Systematic Literature Review Focusing on Datasets, Log Preprocessing, Machine & Deep Learning Approaches and Automated Tool

DEEPALI ARUN BHANAGE[1], AMBIKA VISHAL PAWAR[1], AND KETAN KOTECHA[2]
[1]Symbiosis Institute of Technology, Symbiosis International (Deemed University), Pune 412115, India
[2]Symbiosis Centre for Applied Artificial Intelligence, Symbiosis International (Deemed University), Pune 412115, India

Corresponding author: Ambika Vishal Pawar (ambikap@sitpune.edu.in)

This work was supported by the Research Support the Fund of Symbiosis International (Deemed University), Pune, India.

ABSTRACT Nowadays, reliability assurance is crucial in components of IT infrastructures. Unavailability of any element or connection results in downtime and triggers monetary and performance casualties. Thus, reliability engineering has been a topic of investigation recently. The system logs become obligatory in IT infrastructure monitoring for failure detection, root cause analysis, and troubleshooting. This Systematic Literature Review (SLR) focuses on detailed analysis based on the various qualitative and performance merits of datasets used, technical approaches utilized, and automated tools developed. The full-text review was directed by Preferred Reporting Items for Systematic Reviews and Meta-Analysis (PRISMA) methodology. 102 articles were extracted from Scopus, IEEE Explore, WoS, and ACM for a thorough examination. Also, a few more supplementary articles were studied by applying Snowballing technique. The study emphasizes the use of system logs for anomaly or failure detection and prediction. The survey encapsulates the automated tools under various quality merit criteria. This SLR ascertained that machine learning and deep learning-based classification approaches employed on selected features enable enhanced performance than traditional rule-based and method-based approaches. Additionally, the paper discusses research gaps in the existing literature and provides future research directions. The primary intent of this SLR is to perceive and inspect various tools and techniques proposed to mitigate IT infrastructure downtime in the existing literature. This survey will encourage prospective researchers to understand the pros and cons of current methods and pick an excellent approach to solve their identified problems in the field of IT infrastructure.

INDEX TERMS IT infrastructure monitoring, log analysis, failure detection, failure prediction, machine learning, deep learning, rule-based, NLP, semantic vectorization.

I. INTRODUCTION

In recent years, modern software has been rapidly integrated into organizations and in our daily lives. Also, it turns out to be influential. Most of the applications are intended to be accessible and stable continuously. Any trivial or non-trivial downtime can ignite financial [1] and performance losses. For example, four-hour downtime in Amazon Web Services resulted in a $150 million loss [2]. Thus, it is paramount to maintain IT infrastructure's health to improve its availability and reliability.

In the IT infrastructures, several components and assets are connected and continuously interacting with each other. For this reason, it is always precarious to determine the cause of the failure. System logs are considered as the primary source of data as it records the software's runtime information. Logs generate on the execution of logging statements that programmers write while developing source code. However, making use of enriched log data is challenging because of subsequent reasons. First, the rapidly increasing

The associate editor coordinating the review of this manuscript and approving it for publication was Porfirio Tramontana.

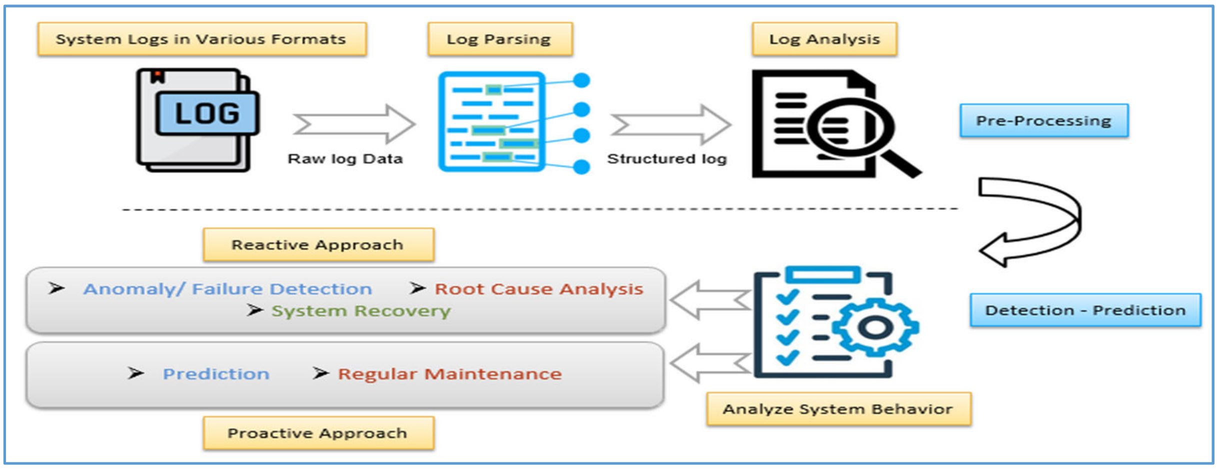


FIGURE 1. **Infrastructure failure detection and failure handling process.**

log volume (for example, an extensive scaling service system can record 50 GB/hour logs [3]). Second, using an open-source platform (for example, GitHub) allows designing a system by many developers [4], multiple development styles resulted in complex logging. Third, change in the nature of logging statements due to the new software versions (hundreds of recent logging statements per month).

Fig. 1 illustrates the steps required in the IT infrastructure failure detection and failure handling process. This process is majorly divided into two parts: a collection of necessary data such as logs, resource-used data, or IT service tickets, and then pre-processing is to reduce the volume. The second part focuses on training and execution of the models for detection and prediction of failures. Detailed discussion on this process has been done in [5] by Bhanage.

In IT infrastructure monitoring, researchers and experts have accomplished ample research in the recent past. As mentioned in the existing literature, failure handling is possible with the help of reactive and proactive approaches [6]. In the study, researchers have explored various types of logs, including RAS log [7], health log [8] event log [9], activity log [10], transactional and operational log [11], etc. Also, log parsing is performed by using frequency pattern mining [12], clustering [13], natural language processing (NLP) techniques [14]. In addition, researchers have explored Machine learning models (SVM, Naïve Bayes, and Random Forest) and Deep learning (RNN, CNN, LSTM, and Bi-LSTM) to detect and predict anomalies or failures in various IT infrastructures.

A. SIGNIFICANCE

In IT infrastructure, many assets and components are connected. They continuously communicate with each other, which generates a massive amount of data. Unavailability of any component or connection in IT infrastructure leads to catastrophic failures and crucial losses [15]. Therefore, it is essential to prevent such failure conditions.

According to Du *et al.* [16], the primary purpose of the log is to record all the executed activities and monitor the status of the IT infrastructure. The system log is also used as the elementary source to identify the problem and troubleshooting [17]. Traditionally developers or administrators were analyzing logs manually to understand the behavior of the system. However, due to the increased complexity and massive data, IT infrastructure monitoring demands automated monitoring [18]. The system logs are enormous and available in an unstructured format. Thus, there is a need to preprocess logs to better understand and retrieve meaningful information from complex log data [14]. Ren *et al.* [19] reveal that log analysis is a comprehensive approach for failure detection, handling, and prediction. Thus it is imperative to prolong the research and utilize the system logs to carry reactive or proactive strategies in order to avoid failures and prevent monetary and productivity losses.

B. MOTIVATION

Hereafter, IT infrastructures will be available everywhere and in a continuously working state [20].

Thus, it is imperative to conduct unbiased research to prepare reliable IT infrastructures and monitor their health [21]. Currently, many popular commercial tools are present in the market for IT infrastructure monitoring. Many IT companies are working for IT infrastructure monitoring using log analysis. Researchers have suggested various new approaches and tools to take care of the continuous availability of IT infrastructure in the recent past. Ample research has been done in IT infrastructure monitoring, but a comprehensive analysis has not been presented until now.

The existing literature on IT infrastructure failure detection and handling techniques focuses on specifications such as log data, pre-processing of the log, machine learning, and deep learning approaches for detection and prediction. After a thorough analysis of existing literature, a comparative study of present tools and techniques is vital. To the best of our knowledge, a limited number of systematic literature reviews are published on the topic. This analysis concentrates on the following key points: availability of datasets, different

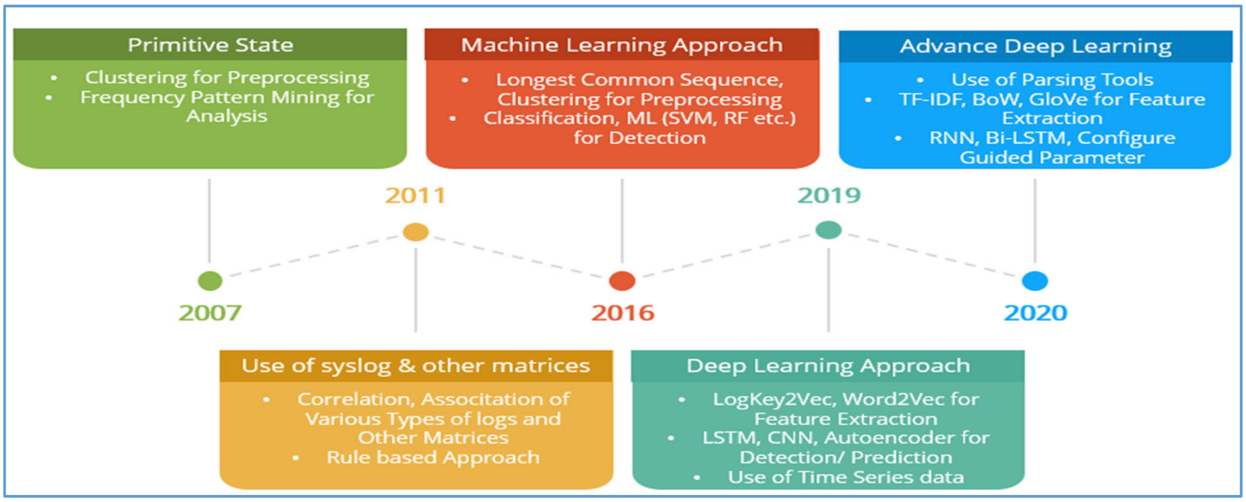

Primitive State
Clustering for Preprocessing
Frequency Pattern Mining for Analysis
Machine Learning Approach
Longest Common Sequence, Clustering for Preprocessing
Classification, ML (SVM, RF etc.) for Detection
Advance Deep Learning
Use of Parsing Tools
TF-IDF, BoW, GloVe for Feature Extraction
RNN, Bi-LSTM, Configure Guided Parameter
2007
2011
2016
2019
2020
Use of syslog & other matrices
Correlation, Associtation of Various Types of logs and Other Matrices
Rule based Approach
Deep Learning Approach
LogKey2Vec, Word2Vec for Feature Extraction
LSTM, CNN, Autoencoder for Detection/ Prediction
Use of Time Series data

FIGURE 2. **Evolution of failure detection and handling techniques.**

technical approaches used to pre-process logs, anomaly or failure detection, and failure prevention to present the study of available literature.

C. EVOLUTION OF THE FAILURE DETECTION AND HANDLING TECHNIQUES

The system logs are rich in information and provide all the details about the activities executed on the IT infrastructure components. Developers and system administrators have been using the system log to identify the IT infrastructure problems and troubleshoot. Also, system experts scrutinize log data manually by considering the different levels of the recorded log data.

Due to the increasing size of log data, automation in log analysis was initiated in 2003 [12] and then further accelerated since 2007. The evolution of failure detection and handling techniques in the studied literature is presented in Fig 2. In the primitive state of the research, the clustering approach was popular for log data pre-processing. Also, log analysis was achieved by using frequency pattern analysis techniques. The researchers started to find the correlation and association between the various types of logs and other metrics to gather further details about the failure, such as the path of failure, causes, component details, etc. Along with correlation, the rule-based approaches were popular for anomaly and failure detection.

Machine learning techniques have been used comprehensively since 2016 because of their classification and prediction proficiency. Random forest, Gaussian NB, Naïve Bayes, Support Vector Machine (SVM) are immensely utilized machine learning algorithms for anomaly and failure detection. In the year 2019, researchers have begun to use Word2Vec, TF-IDF, GloVe, etc. NLP techniques for feature extraction considering log as standard text data. Due to the increase in log data size, deep learning techniques such as RNN, CNN, LSTM, Bi-LSTM have been applied to train detection and prediction models. Many researchers employed the Auto-Regressive Integrated Moving Average (ARIMA) time series technique to predict time series log data.

D. PRIOR RESEARCH ON SYSTEMATIC LITERATURE REVIEW

In the existing literature, reviews are carried out on log abstractions, log clustering, anomaly detection using deep learning techniques, log data quality analysis, log data for troubleshooting, etc. However, these surveys do not cover all the elements such as dataset, techniques, approaches, research gaps, etc., they concentrate on a certain part of the IT infrastructure monitoring research.

El-Masri *et al.* [22] 2020 published the SLR of automated log-abstraction techniques (ALAT). In this review, the authors evaluated 17 automated log abstraction techniques on seven aspects: mode, coverage, delimiter independence, efficiency, scalability, system knowledge independence, and parameter tuning efforts.

Cyber-attack can be one of the reasons for IT infrastructure failure; for this, we have considered the survey of log clustering approaches in cybersecurity applications. Landauer *et al.* [23] in 2020 illustrated clustering techniques, anomaly detection, and evaluation aspects in cybersecurity application with the help of assessing 50 approaches and two non-academic solutions. The authors also presented a clustering approach selection tool based on the analysis done in the survey. This tool provides ranking to the approaches by taking the ability to fulfill objectives and visualize results on the PCA plot.

Yadav *et al.* [24] in 2020 published a survey on anomaly detection using deep learning techniques. The survey focused on NLP-based approaches for feature extraction, whereas machine learning and deep learning methods for anomaly detection using log data. Das *et al.* [25] presented a systematic mapping analysis in 2020 to discuss the general approaches

TABLE 1. Research questions and objectives.

Research Questions		Objectives
RQ1: How are log entries valuable for troubleshooting the failure?	1.	Study existing work on troubleshooting the failure using log data and compare with other approaches
	2.	Study the features of logs to utilize them for troubleshooting the problems.
RQ2: What are the different IT Infrastructures and logs used to perform research experiments?	1.	Explore the literature to study various IT infrastructures and log data utilized for experimentation.
	2.	Check the availability of log datasets that are released by researchers in literature for further study.
RQ3: What is the performance of the different approaches for anomaly and failure detection in IT Infrastructure?	1.	Analyze various anomaly and failure detection approaches based on datasets used, techniques, algorithms, features extraction approaches, performance, etc.
	2.	Conduct comparative study and list down the capable options
RQ4: What are the various existing techniques available to prevent and predict failures in IT infrastructure monitoring?	1.	Review and state different failure prevention methods investigated in the literature.
	2.	Analyze various failure prediction approaches based on datasets used, techniques, algorithms, features extraction approaches, performance, etc.
	3.	Conduct comparative study and list down the capable options
RQ5: What are the different state-of-the-art tools and techniques used for log monitoring and analysis?	1.	Explore automated tools for parsing, log analysis, failure or anomaly detection, prediction in IT infrastructure.
	2.	Assess state-of-the-art automated tools in accordance with techniques and merit.
RQ6: What are the distinguished limitations of existing literature?	1.	Conduct a rigorous investigation of selected scholarly articles to identify potential research gaps.
	2.	Suggest future directions to forthcoming researchers

for failure prediction using logs. In the survey [26], Shilin *et al.* in 2020 address the questions such as ''How to write logging statements automatically'', ''How to compress and parse log'', ''How to use the log to detect, predict, facilitate diagnosis of the failure''. This survey presents various challenges in the studied literature but fails to provide a comparative analysis.

Bhanage [5] in 2021 categorized literature into three major groups: log pre-processing, anomaly & failure detection, and failure prevention; in this study, authors furnished the meta-analysis contingent on infrastructure used, dataset utilized for analysis, category of work, and methodology used. The authors also enumerated the automated tools for log parsing, log analysis, anomaly or failure detection, prediction, and recovery of IT infrastructure.

Our SLR examines the existing approaches, methodologies, and tools relying on various merit criteria (mode, availability, industrial utilization, and accuracy). The SLR endeavors to open a window of opportunities for forthcoming researchers in the area of IT infrastructure monitoring. In this comprehensive study, we strive to emphasize on consecutive aspects: the availability of datasets for various types of infrastructure, methodologies utilized for detection and prediction, and publicly available automated tools for log pre-processing technical approaches for detection with evaluation metrics and failure prevention techniques.

E. RESEARCH QUESTIONS

This paper attempts to conduct an exhaustive review of the existing literature on IT infrastructure monitoring techniques. The subsequent research questions are accosted in the study. We are facilitating a better understanding of the current literature by answering these questions. Furthermore, the answers to these research questions demonstrate the effectiveness of methods to lead the systematic literature review. Table 1 specify the list of research questions and the objectives of the defined research question.

F. OUR CONTRIBUTION

The systematic literature review emphasizes the current study carried out in IT infrastructure monitoring to maintain the health of IT infrastructure components. In this rigorous analysis, we explored the various tools and techniques used to handle the failure conditions. It also focuses on the miscellaneous frameworks, methodologies, approaches developed and pursued by several researchers. The comparative analysis and the concluding remarks on various components of the literature study are provided as an outcome of answers to the research questions. The systematic literature study has scrutinized different tools and techniques based on results that pinpoint the vital research gaps.

The rest of the paper is organized as follows. The methodology and framework exerted to extract and scrutinize scholarly articles from various databases for review have been discussed in Section 2. Section 3 illustrates the impact of scholarly publications in the existing literature of IT infrastructure monitoring. Section 4 describes the architecture of the proposed system. Section 5 accomplished a comprehensive discussion on the experimentation process and derived results. Section 6 discusses the research questions, which further leads to a systematic literature review. Section 7 states the paper's limitations, discusses future directions. Section 8 exchanges view on concluding remarks.

II. METHODOLOGY FRAMEWORK FOR SYSTEMATIC LITERATURE REVIEW

A systematic literature review of the available literature was undertaken to find the answers to the proposed research

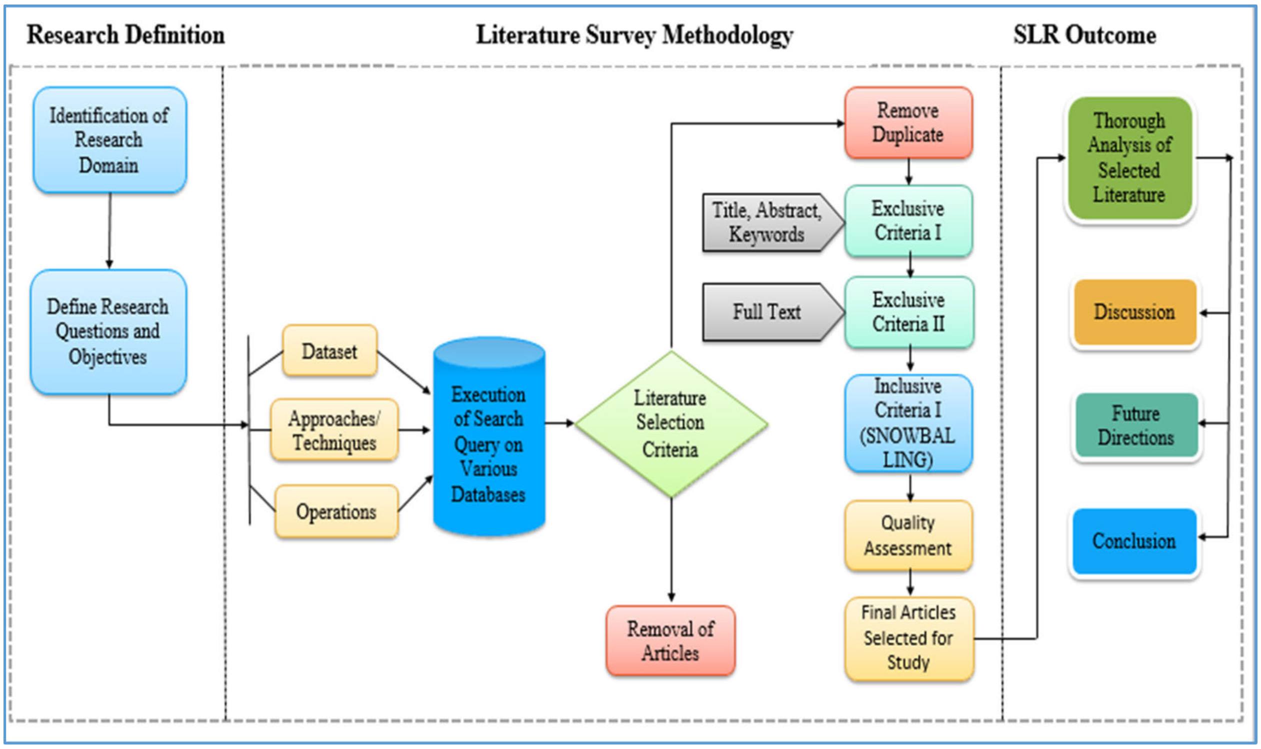

Research Definition
Literature Survey Methodology
SLR Outcome
Identification of Research Domain
Define Research Questions and Objectives
Dataset
Approaches/ Techniques
Operations
Execution of Search Query on Various Databases
Literature Selection Criteria
Remove Duplicate
Title, Abstract, Keywords
Exclusive Criteria I
Full Text
Exclusive Criteria II
Inclusive Criteria I (SNOWBALLING)
Quality Assessment
Final Articles Selected for Study
Removal of Articles
Thorough Analysis of Selected Literature
Discussion
Future Directions
Conclusion

FIGURE 3. Systematic literature review (SLR) methodology.

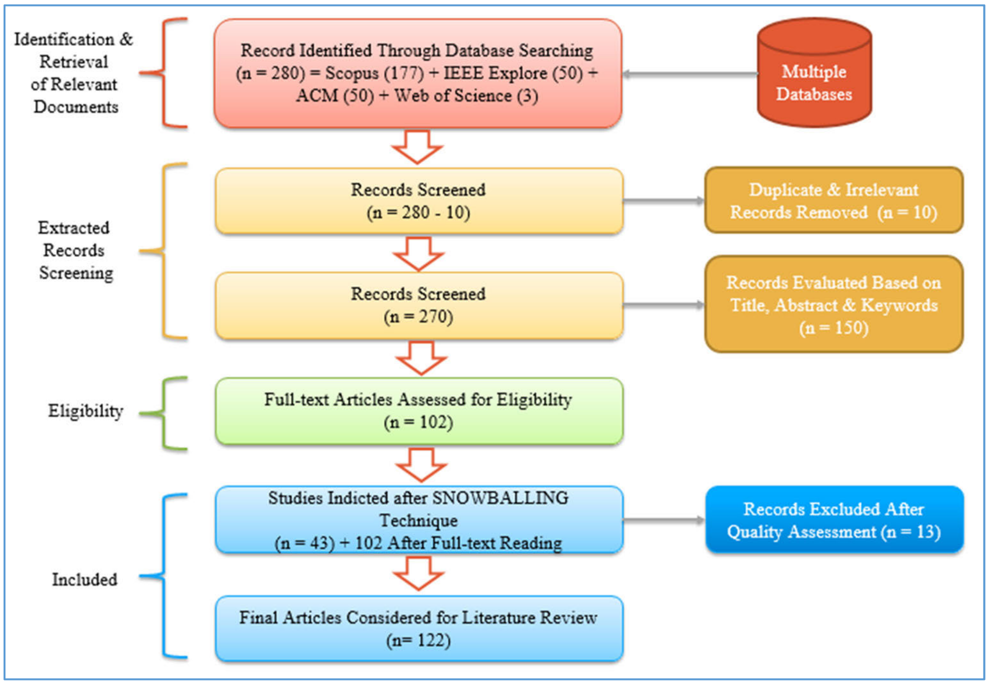

Identification & Retrieval of Relevant Documents
Record Identified Through Database Searching (n = 280) = Scopus (177) + IEEE Explore (50) + ACM (50) + Web of Science (3)
Multiple Databases
Extracted Records Screening
Records Screened (n = 280 - 10)
Duplicate & Irrelevant Records Removed (n = 10)
Records Screened (n = 270)
Records Evaluated Based on Title, Abstract & Keywords (n = 150)
Eligibility
Full-text Articles Assessed for Eligibility (n = 102)
Included
Studies Indicted after SNOWBALLING Technique (n = 43) + 102 After Full-text Reading
Records Excluded After Quality Assessment (n = 13)
Final Articles Considered for Literature Review (n= 122)

FIGURE 4. Process of appropriate publications selection for comprehensive analysis.

questions and objectives. This process will shed light on the potential research gaps and the challenges encountered in studied research areas and discuss viable solutions. Comprehensive guidelines suggested by Kitchenham *et al.* [27] were adapted to accomplish the thorough systematic literature review. Table 2 presents the five elements of PICOC (Population, Intervention, Comparison, Outcome, and Context) for framing the searchable questions suggested by Kitchenham.

TABLE 2. PICOC (population, intervention, comparison, outcomes, context).

Criteria	Discussion
Population	Various IT infrastructures include distributed systems, supercomputers, operating systems, Services, networks, server applications, mobile systems, and standalone software.
Intervention	Tools, techniques and approaches applied for log processing, anomaly detection, failure detection, anomaly prediction, failure prevention.
Comparison	Comparison of various anomaly or failure detection and prediction techniques. Performance of different machine learning and deep learning approaches. Availability and performance comparison of state-of-art automated tools for log parsing and log analysis.
Outcomes	The systematic literature review, based on various tools, techniques and approaches for failure detection and prevention.
Context	The structured analysis to consolidate academic and IT industry research, comparison of tools, techniques and approaches. Study of recent trends and future directions.

Fig. 3 illustrates the process and methodology used for the systematic literature review. The research domain is identified for the systematic literature review, followed by defining the research questions and objectives. The significant material is collected based on the dataset, approaches, techniques, and operations to study and answer the research questions. Intended search query executed on various repositories such as Scopus, ACM, IEEE and Web of Science to collect relevant scholarly publications. Then inclusive and exclusive selection criteria are applied to select the most appropriate publications for thorough analysis. The analysis of studied literature is epitomized through discussion on answers to research questions, future directions for forthcoming researchers and a conclusion to state the concluding remarks.

A. RESEARCH PUBLICATIONS COLLECTION CRITERIA

Scholarly articles were collected from various databases like Scopus, ACM, IEEE, Web of Science. We delineated a search query using pertinent keywords such as "system log or event log," "log Analysis," "failure detection or failure prediction," "machine learning or deep learning," etc., to retrieve the related articles. The listed prominent keywords have been utilized by Bhanage and Pawar [28] to collect the information for bibliometric analysis. The same search query was executed on multiple databases to retrieve appropriate publications. The count of the extracted publications is presented in Table 3.

B. INCLUSIVE AND EXCLUSIVE CRITERIA

As stated in Table 4, inclusive and exclusive criteria were applied to acquire the most relevant scholarly articles for systematic literature review. Utilization of various IT infrastructures, different approaches or methodologies used for detection or prediction, and multiple preprocessing techniques parameters applied to select articles for further analysis.

C. PUBLICATION COLLECTION RESULT

In the publication selection results, 177 publications found in the Scopus database, ACM and IEEE, identified 50 articles, whereas Web of science extracted only 3. All research articles were investigated thoroughly and categorized into three groups based on the work's intent. Detailed discussion is done on log preprocessing, anomaly or failure detection, and failure prevention types in the forthcoming sections.

D. SELECTION PROCESS AND RESULT

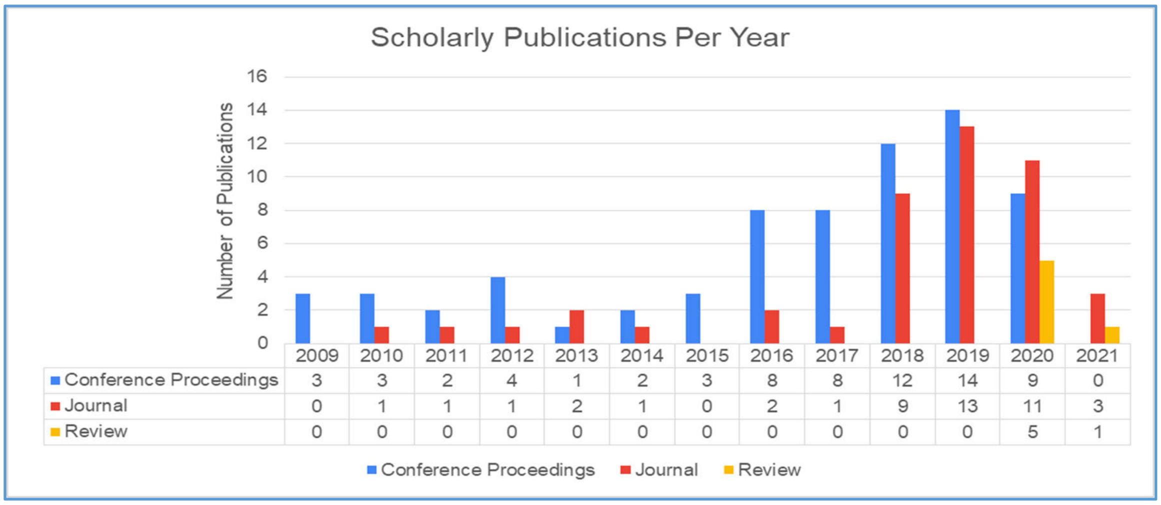
Fig. 4 exhibits the process followed while selecting a publication for a detailed study by effectuating inclusion and exclusion criteria. A total of 280 scholarly publications were extracted from various repositories, as indicated in Table 3. The list was reduced to 270 entries by removing equivalent and irrelevant articles. The 270 publications probed through the title, keywords, article's abstract, and 150 scholarly articles were selected for the analysis. In addition to databases, we applied the backwards snowballing technique to identify more articles [29]. In the backward snowballing approach, authors track the references list of the primarily selected papers. Most relevant articles from the references are shortlisted based on the inclusive and exclusive criteria stated in Table 4. Supplementary 43 articles were added for study using Snowballing technique. Finally, 122 articles were selected by excluding 13 articles after quality assessment. Besides scholarly articles, we also referred to a few web links to study and gather information related to ITSM concepts and commercial tools. Fig. 5 presents the contribution of study material as per type's publication in the systematic literature review.

E. QUALITY ASSESSMENT CRITERIA

Quality assessment criteria were applied to select the particular scholarly publications to effectuate the systematic literature survey on IT infrastructure monitoring research. The desired quality articles must significantly contribute to answering the research questions.

Following are the quality measures referred to shortlist scholarly articles:

- *IT infrastructure*: The article must be focused on IT infrastructure monitoring by employing log data and other resource-related metrics.
- *Datasets*: The articles emphasized the various components of datasets such as type of dataset, infrastructure utilized, time frame and data size.
- *State of the art tools*: The articles discussed the existing automated tools for log parsing and analysis and presented details such as the technique used, mode, availability, industry utility, and accuracy.
- *Classification based approaches*: The articles studied machine learning, or deep learning-based classification



	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021
🟦 Conference Proceedings	3	3	2	4	1	2	3	8	8	12	14	9	0
🟥 Journal	0	1	1	1	2	1	0	2	1	9	13	11	3
🟨 Review	0	0	0	0	0	0	0	0	0	0	0	5	1

FIGURE 5. Type of publications per year from retrieved results.

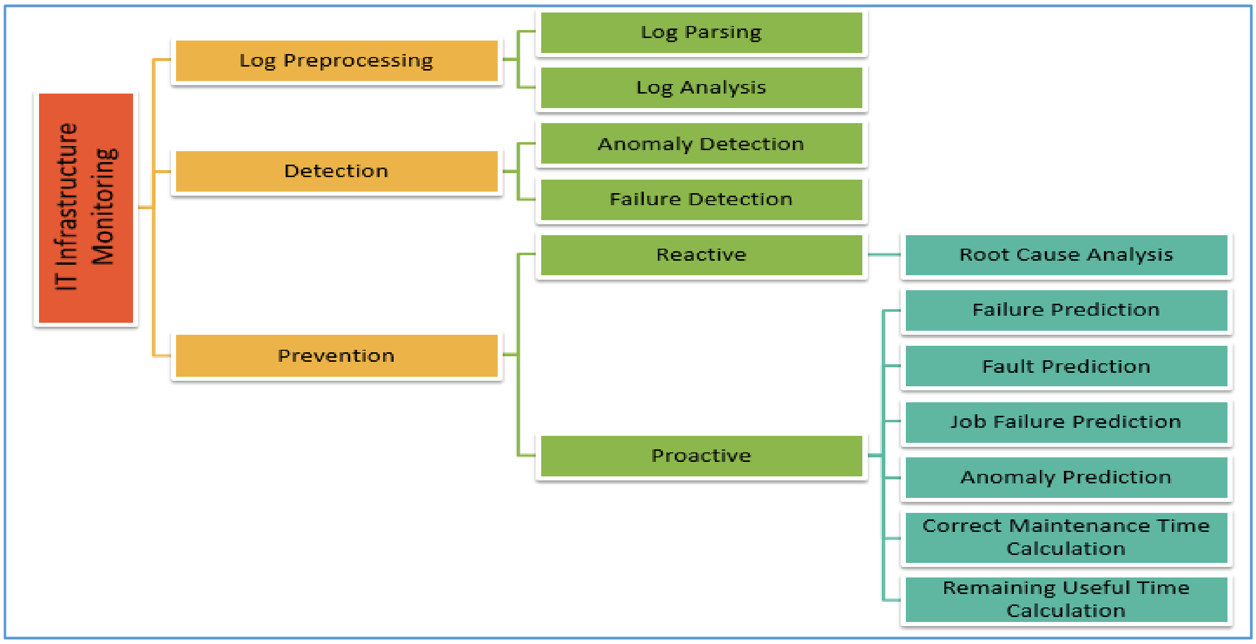


FIGURE 6. Classification of studied scholarly articles from collected literature according to purpose.

approaches for anomaly or failure detection and prediction. Moreover, the article provided the information of dataset, technical approach, preprocessing or feature extraction techniques and metrics used for evaluation.

- *Data validation*: The articles particularly commented on the findings and results considering the stated objectives and expected outcomes.

F. DATA EXTRACTION AND SYNTHESIS FOR SYSTEMATIC LITERATURE REVIEW

Table 5 presents the data collection and synthesis to conduct the systematic literature review by answering the formulated research questions. The table furnishes the particulars challenges encountered while finding the answers to the research questions, contents extracted for meticulous study from selected scholarly articles, and the study performed by us. The survey conducted by us to satisfy the answers to research questions is discussed in detail in the upcoming sections of the paper.

III. IMPACT STUDY OF SCHOLARLY PUBLICATIONS

A. DATA EXTRACTION FROM EXISTING LITERATURE

Fig. 6 shows the classification of studied scholarly publications. The publications were classified by perusing the title, abstract, keywords, and full text of the selected publication for analysis. According to the work's purpose, the articles

TABLE 3. Literature database, search query and count of extracted publications.

Database	Search Query	Number of Publications Found
SCOPUS	ALL ((("Events" OR "System log" OR "Event log") AND ("log Analysis") AND ("failure detection" OR "failure prediction" OR "Fault Tolerance") AND ("Machine Learning" OR "Deep Learning")))	177
ACM	[[All: "events"] OR [All: "system log"] OR [All: "event log"]] AND [All: "log analysis"] AND [[All: "failure detection"] OR [All: "failure prediction"] OR [All: "fault tolerance"]] AND [[All: "machine learning"] OR [All: "deep learning"]]	50
IEEE	((("All Metadata": Events) OR ("All Metadata": System log) OR ("All Metadata": Event log)) AND ("All Metadata": Log analysis) AND (("All Metadata": Failure Detection) OR ("All Metadata": Failure prediction) OR ("All Metadata": fault tolerance)) AND (("All Metadata": Machine Learning) OR ("All Metadata": Deep Learning)))	50
Web of Science	TOPIC: ((("Events" OR "System log" OR "Event log") AND ("log Analysis") AND ("failure detection" OR "failure prediction" OR "Fault Tolerance") AND ("Machine Learning" OR "Deep Learning"))) Timespan: All years. Databases: WOS, KJD, RSCI, SCIELO. Search language=Auto	3

Accessed on 20th June 2021

TABLE 4. Inclusive and exclusive criteria for selection appropriate scholarly articles.

Criteria Number	Content	Inclusion Criteria	Exclusion Criteria
1.	Infrastructure, log data, dataset	The article must focus on IT infrastructures to collect the log dataset for monitoring and provide details about log datasets available for further research.	The articles focus on non-IT infrastructure. Research not monitoring log data for the study.
2.	Approaches used for anomaly or failure detection	Selected articles are essential to contain information about anomaly or failure detection approaches in IT infrastructure components.	Articles focus on anomaly or failure detection but do not comment on the accuracy.
3.	Techniques or methodologies used to prevent failure	The articles should emphasize the failure prevention techniques either reactively or proactively.	The articles that are not discussing the failure prevention techniques with adequate accuracy.
4.	Publication year and other details	The articles were published from 2009 till early 2021, carrying details such as journal name, volume, issue etc.	The articles do not provide precise details about journal name, volume, issue etc.
5.	Research questions	The articles must contribute to determining the answer to a minimum of one research question.	The articles were not relevant to the contents of the research questions.

related to IT infrastructure monitoring are studied carefully and classified into three categories: pre-processing, detection, and prevention. Pre-processing is further partitioned into log parsing and log analysis. Detection of anomaly and failure conditions targeted in the studied literature. Prevention techniques are categories into reactive and proactive approaches. The reactive process takes place after the occurrence of a failure condition. Whereas, proactive approach predicts the error conditions before it takes place.

For RQ1, the importance of log data for troubleshooting is studied based on the nature of logs. Various types of datasets, their availability, and the dataset size were discussed to answer RQ2. Anomaly and failure detection approaches, techniques, and performances these parameters evaluated to answer RQ 3. Reactive and proactive strategies were studied and summarized to extract the answer to RQ4. For RQ5, various state-of-the-art tools are analyzed based on techniques used, mode, availability, and industrial utility and accuracy parameters. All the selected articles are studied cautiously to discover distinguished limitations for RQ6.

All the categories and targeted evaluation points are discussed in detail in the upcoming sections of the paper.

B. DATASETS

1) TYPES OF DATASETS

Studies were performed on various infrastructures such as distributed systems, supercomputers, operating systems, mobile systems, server applications, and standalone software in the literature. Fig. 7 demonstrates the different types of infrastructures and systems explored in the systematic literature review.

2) AVAILABILITY OF DATASETS

To make a log dataset available for study is a challenging task. Log data provides all the details about the execution of the infrastructure components, and the misuse of this data may cause serious problems. Thus, log data are not readily available for use or experimentation due to strict business policies and confidentiality issues. Few sample logs are collected from the existing literature and released for research studies in academia. Table 6 furnishes the list of the infrastructure type, system, infrastructure, dataset type, time frame, and size of the collected data. Zhu *et al.* [30] in 2019 released a log dataset repository of different 16 types of systems on loghub [31]. Out of these few datasets

TABLE 5. Data collection and synthesis of selected scholarly articles to answer the research questions.

Research Questions	Challenges in Finding Answers	Contents Extracted from Selected Scholarly Articles	Study Performed
RQ1: How are log entries valuable for troubleshooting the failure?	• Security concerns due to a confidential and rich source of data • Unavailability of log data	• Study of logging statement in view of elements of recorded logs • Collection of datasets with references, type of data, size of data etc.	• We examined logs, patterns and information which help in troubleshooting.
RQ2: What are the different IT Infrastructures and logs used to perform research experiments?	• Variations in logging statements • Logs in different formats • Unstructured logs	• Study of different types of IT infrastructures • Retrieve the log template with the help of log parsers • Conversation of logs from unstructured to structured format using log parsing • Representation of log records in more understandable form with the help of log analysis	• Sample IT infrastructures categorized according to type. • 16 log parsing tools studied, analyzed and summarized in regard to 4 merit criteria. • 8 log analysis tools were studied, analyzed and summarized in regard to 4 merit criteria.
RQ3: What is the performance of the different approaches for anomaly and failure detection in IT Infrastructure?	• No generalized solution • The massive volume of log data	• Results of anomaly and failure detection approach for different IT infrastructures • The effect of preprocessing or feature extraction to improve the accuracy of detection • Rule-based, correlation-based and classification-based approaches were explored considering the infrastructure, dataset, evaluation matrices etc.	• Anomaly and Failure Detection Approaches studied carefully and summarized in tabular form based on infrastructure and dataset used, technical approaches, preprocessing or feature extraction, algorithms and performance using various metrics
RQ4: What are the various existing techniques available to prevent and predict failures in IT infrastructure monitoring?	• Unavailability of historical data • Typical failure pattern not present	• Study of reactive and proactive approaches to avoid failure condition • Logs in time series data format might be effective to identify the pattern • Survey of advanced and sophisticated deep learning techniques	• Different types of prediction strategies were studied carefully and summarized in tabular form based on infrastructure and dataset used, technical approaches, preprocessing or feature extraction, algorithms and performance using various metrics
RQ5: What are the different state-of-the-art tools and techniques used for log monitoring and analysis?	• Preparing environment set up for tool execution • System dependent solutions	• commercial and academic Log parsing and log analysis tools explored from literature and web links	• Environment set up prepared, and open-source parsing tools are executed to check the accuracy • Drain parser executed for different types of datasets and check the performance
RQ6: What are the distinguished limitations of existing literature?	• Log formats are not consistent. Literature does not have consistent findings	• 122 scholarly articles analyze exhaustively based on the constraints and future work	• On comprehensive analysis, 10 research gaps were identified in the existing literature • Potential future directions provided for forthcoming researchers

(for example, HDFS, Hadoop, BGL) are utilized and released by the previous researcher. In contrast, other datasets were collected from the Zhu *et al*. authors' lab environment. The component failure log data from various extensive production systems are accessible on the computer failure data repository (CFDR) [32]. Los Alamos National Lab (LANL), HPC cluster, Internet services cluster, Cray systems, and Blue Gene/p system's different types of logs are available to accelerate the research on system reliability. Another research study [33] presented the Apache log files that record and store internet search traffic for EDGAR filings through SEC.gov from 14th February 2003 to 30th June 2017. Cotroneo *et al.* in 2019 [34] executed an empirical analysis of software failure in the OpenStack cloud system. The failure dataset with injected faults, the workload, the failure effect at the user and system side, and error logs used for study and release for further research [35]. Apart from this, there are log datasets collected for cybersecurity research. SecRepo [36] holds a list of security data like threat feeds, malware, system, network, etc.

3) CHALLENGES WITH LOG DATASE

Fig. 8 presents the challenges in the availability and utilization of log datasets.

- Unavailability of Data due to Sensitivity

Log data carries all the details about the system such as resources involved, event records, sequence of performed activities and other information. That's why these rich logs are considered sensitive data. Misuse of such sensitive information may result in security and different types of issues.

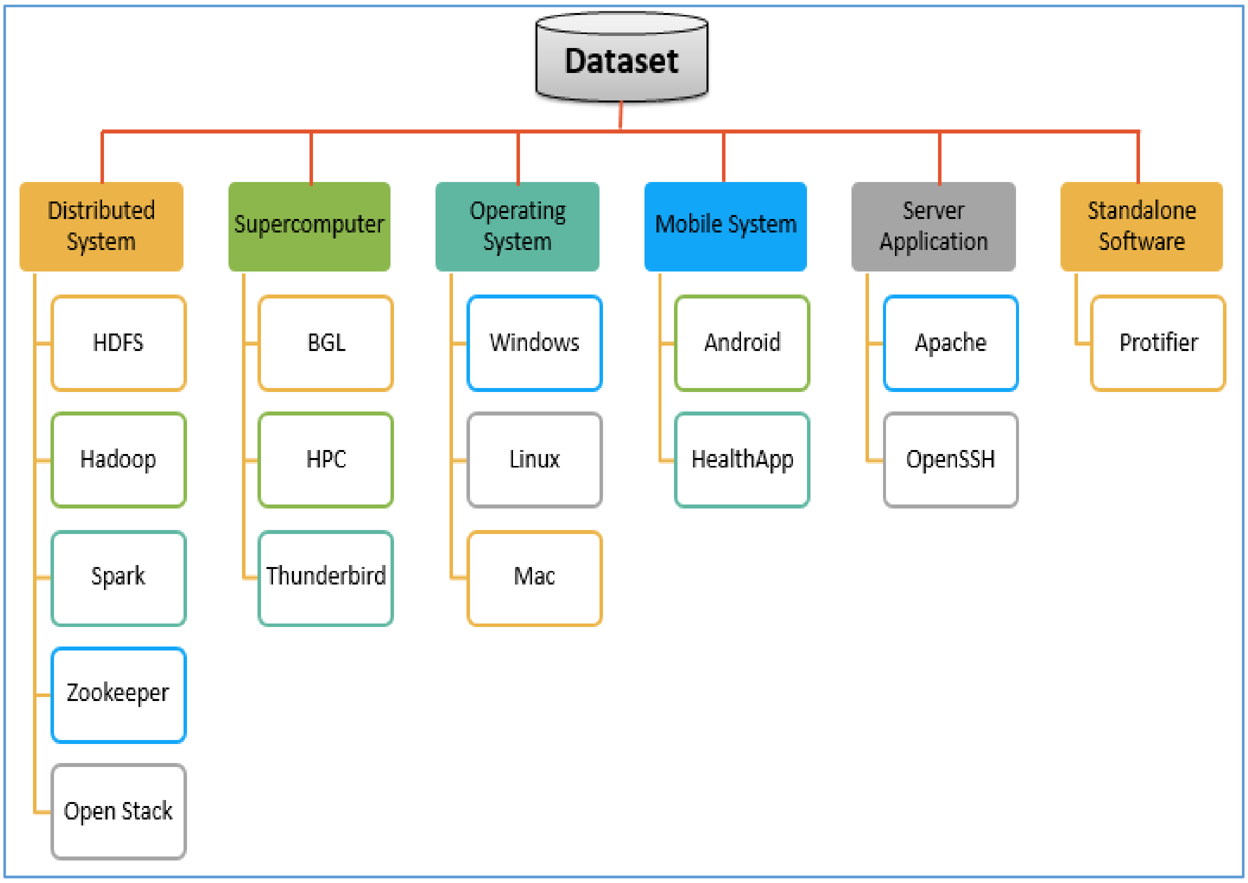


FIGURE 7. Various types of infrastructure studied to collect datasets and used for anomaly or failure handling in systematic literature review.

This is the reason why logs and event records are not available publicly easily. The unavailability of logs for experimentation is the biggest challenge faced by researchers.

- Huge Data Size

As IT infrastructure's complexity and execution increase rapidly, massive log data is getting generated every second. According to literature, continuously functioning infrastructure can record approximately 50 GB/hour logs [3]. This gives rise to the increase in the log volume. Making use of enormous volume data for experimentation is challenging by considering problems in the management of data, finding & fixing the quality issues, data integration, controlling big data environment etc.

- Different data formats

Logs are recorded on the execution of logging statements written by developers during software development. There is no fixed format or template present for logging statements. Each developer may follow their logging statement style bearing in mind the required contents. Many development styles are accessible in the software, as the use of open-source platforms is increasing expeditiously. Also, many source codes are available on GitHub for reuse [4]. These multiple contributions with various development styles give rise to log data generation with different formats, which causes troubles in the development of standard data analysis processes.

- Imbalanced Data

Historical log records will be collected for experimentation through various IT infrastructures. Generally, IT infrastructures operate under a normal state; thus, it is severe to collect anomalous records. According to Yan *et al.* [47], there is a need to handle the imbalanced data to improve fault detection and diagnosis results. The authors applied a Generative adversarial network (GAN) to convert imbalanced training data to balanced training data.

- Inconsistent log generation

Inconsistent logs are getting generated due to the change in the nature of logging statements. Various developers develop different software versions and follow their own writing styles. This results in inconsistent logging statements followed by unstable logs.

- System Dependent Data

In any IT infrastructure, the nature of components and their communication varies based on their utility. As of now, no standard rules or conventions are present to write logging statements. Each system has separate ways to write logging statements and record the different types of details. Therefore it generates various types of logs. Multiple researchers have studied different kinds of systems independently in the literature because of the diversity in log format.

TABLE 6. Details of available datasets for use in existing literature.

Ref	Infra Type	System	Infrastructure	Dataset type	Time Frame	Data Size
[37]	Distributed system	Hadoop	Distributed system	Hadoop distributed file system log	38.7 hours	1.47 GB
[38]		HDFS	Distributed system	Hadoop distributed file system log	-	48.61 MB
[14]		Spark	Distributed system	Spark job log	-	
[39]		ZooKeeper OpenStack	Distributed system	ZooKeeper service log OpenStack software log	26.7 Days	9.95 MB
[14]		Spark	Distributed system	Spark job log	-	60.01 MB
[38]	Supercomputer	BGL HPC	Supercomputer	Blue Gene/L supercomputer log High-performance cluster log Thunderbird supercomputer log	214.7 days	708.76 MB
[40]		Thunderbird	Supercomputer	Blue Gene/L supercomputer log High-performance cluster log Thunderbird supercomputer log	-	32.00 MB
[30]		BGL HPC	Supercomputer	Blue Gene/L supercomputer log High-performance cluster log Thunderbird supercomputer log	244 days	29.60 GB
[41]		LANL	22 HPC cluster systems	Records of cluster node outages, workload logs, and error logs	December 1996 to November 2005	-
[42]		HPC	765-node HPC cluster with 64 filesystem nodes	Hardware replacement log	August 2001 thru May 2006	-
[43]		Thunderbird	5 HPC systems with 512 to 131072 processors	Event log	Between 215 and 558 days in 2004 - 2006	1.9 GB
[44]		Spirit	5 HPC systems with 512 to 131072 processors	Event log	Between 215 and 558 days in 2004 - 2006	864 MB
[45]		Liberty	5 HPC systems with 512 to 131072 processors	Event log	Between 215 and 558 days in 2004 - 2006	641 MB
[46]		Blue Gene/P	Blue Gene/P	RAS log	Jan 09 - Aug 09	-
[38]	Operating System	Windows	Operating System	Windows event log	226.7 days	26.09 GB
[40]		Linux	Operating System	Linux system log	263.9 days	2.25 MB
[40]		Mac	Operating System	Mac OS log	7.0 days	16.09 GB
[38]	Mobile System	Android	Mobile System	Android framework log	-	3.38 GB
[40]		Health App	Mobile System	Health App log	10.5 days	22.44 MB
[38]	Server Application	Apache	Server	Apache Server error log	263.9 days	4.90 MB
[40]		OpenSSH	Server	OpenSSH server log	28.4 days	70.02 MB
[40]	Standalone Software	Proxifier	Software	Proxifier software log	-	2.42 B

Distributed systems, supercomputers, operating systems, mobile systems, server applications, standalone software, and any other system carry different types of logs as described in Table 6.

C. LOG PRE-PROCESSING APPROACHES AND TOOLS

The collected log is always in an unstructured, duplicate, and ambiguous format. Log pre-processing is foremost crucial before transmitting it for analysis. The pre-processing includes three steps 1) Log filtering, which removes the duplicate and noisy data, 2) Log parsing, which converts unstructured log to a structured format; and 3) Log analysis which visualizes the log in a more readable and understandable format.

Hassani *et al.* [48] claimed that sometimes log messages are unreliable. They may hold errors such as improper log messages, lacking logging statements, unsatisfactory log level, log archives structure problems, runtime problems,

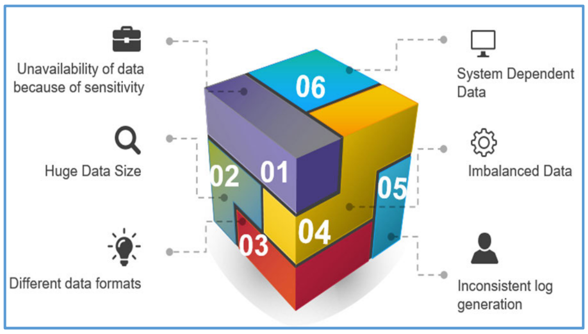


FIGURE 8. Challenges studied in systematic literature review with log data.

overpowering logs, and log files library alterations. Thus, before pre-processing log data, need to validate its quality.

Fig. 9 has listed the techniques and approaches used for log data pre-processing on critical analysis of scholarly publications. As stated in Fig. 9, semantic value similarity, duplicate removal, and adaptive similarity are applied to remove duplicate and redundant data from the log. The log filtering step is considered a data cleaning process. Frequency pattern mining, clustering, heuristics, and longest common subsequence are the commonly used approaches for log parsing. For log analysis clustering (DBSCAN, same level of log), sematic techniques (for example, the appearance of words, text mining), and semantic value similarity (friend of a friend) techniques employed by researchers in the literature.

1) LOG FILTERING

Irrelevant and redundant data generally leads considerable noise to feature extraction and affects the accuracy of the analysis. Log filtering is removing duplicate or unwanted data and reducing the size of the logs. Log filtering is possible with the help of the following techniques: semantic value similarity, duplicate removal, and adaptive similarity. In the literature, Di *et al.* [49] conducted duplication filtering prior to log analysis as ras log has numerous same messages. Whereas Liu *et al.* [50] proposed a filtering threshold to categorize the clusters into normal and anomaly candidates, thus author can discard regular events and concentrate on others to analyze. In addition to this, oliner and stearley [46] claimed that filtering switches alert distribution drastically by removing duplicate alerts within the last 'T' seconds. Ren *et al.* [20] removed stop words and punctuation from removing redundant event data in distributed cluster systems.

2) LOG PARSING

After log collection, it is imperative to be parsed before sending it for further analysis. In the log, some part is constant (written by the developer), and some are dynamic (update at runtime). The primary objective of the parser is to recognize the persistent and variable data. The contact part of the log represents the event template. Thus, the output of the log parser gives log data with the following contents: timestamp, level, component, event template, and critical parameter. In the systematic literature review, we explored 16 automated log parsing tools. To evaluate these parsers, we focus on techniques used and four merit criteria: mode, availability, industrial utilization, and accuracy, as shown in Table 7. SLCT [12] stands for Simple Logfile Clustering Tool; this tool is based on the novel log clustering approach to identify the log files' patterns. Similarly, LFA [51] also works on the same clustering technique to abstract log lines and derive event types. LogCluster [52] is similar to SLCT, but this performs better on log messages with flexible lengths. LKE [53] proposed a novel algorithm to get critical log messages from the Hadoop and SILK system's unstructured log data. The proposed algorithm was further used in anomaly detection, such as workflow errors and low performance of the selected system. SHISO [17] can continuously dig and refine the log template on real-time system logs of OpenSSH except for any prior knowledge by applying a structured tree concept. AEL [54] is the tool designed to monitor the execution of applications using execution logs where log lines are expressly not for monitoring purposes. Extensive enterprise applications were considered to check the tool's performance, and derived results gives 90% precision and 98.4% recall, respectively. LenMa [55] is an online template generator tool with one-pass template mining techniques. It carries out the classification of log messages based on the length of words of each message and forms clusters for same-length messages to identify unique system log message patterns.

OILog [14] proposed extracting keywords from the unstructured log and design log templates by applying a multilayer dynamic PSO algorithm (MDPSO). The tool can pull out the keywords from a real-time and new log with higher efficiency than the existing four tools. LogParse [56] framework works on word classification problems instead of template generation to discover the features of the template and variable words. It also works efficiently on new types of generated logs. The Drain [57], an online parsing tool, is based on a directed acyclic graph and maintains log groups through the tree's leaf nodes. This tool gives 99.9% accuracy on BGL, HDFS, and Zookeeper data sets over LKE, IPLoM, SHISO, and Spell parsers. POP [39] operate on parallel processing; this uses distributed computing to speed up the parsing process of large scale logs. POP reduces the parsing time as compared to other parsers (200 million log messages in only 7 mins).

Spell [58] supports parallel implementation, which helps to accelerate the parsing process. Spell utilizes specialized data structures such as inverted trees and prefix trees. LogMine [13] work efficiently on heterogeneous log messages generated by various systems. It was implemented in the map-reduce framework to extract high-quality patterns by processing millions of log messages in a second. Craftsman [59] is an online parsing tool that applies prefix-tree and frequent patterns techniques for template matching. But this tool fails to merge similar templates effectively.

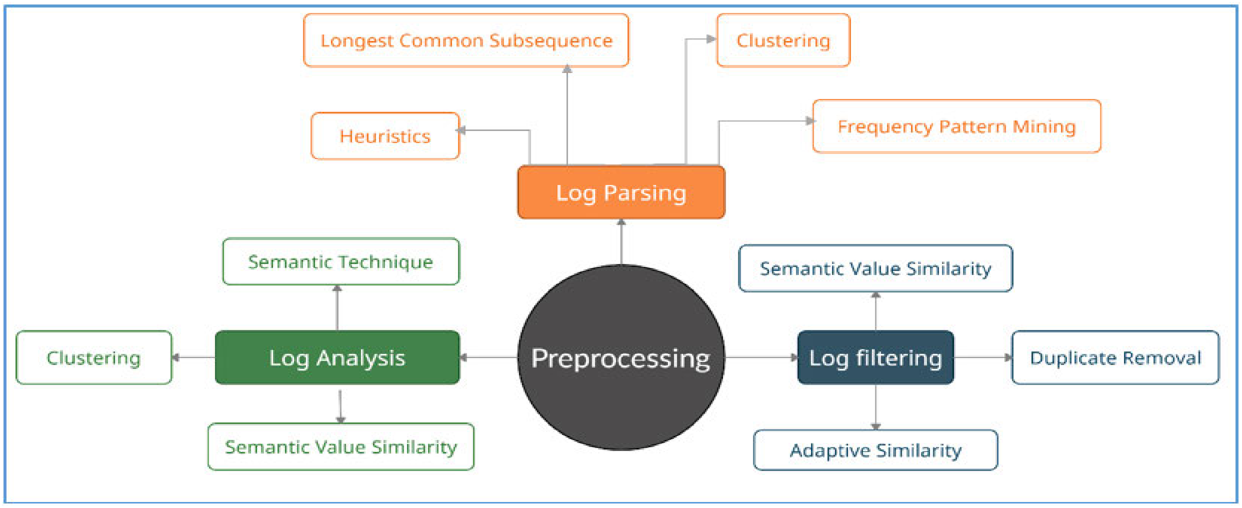


FIGURE 9. Log preprocessing techniques and approaches studied in systematic literature review.

TABLE 7. Log parsing tools studied in the systematic literature review.

Ref	Year	Tool Name	Technique	Merit Criteria			
				Mode	**Availability**	**Industrial Utility**	**Accuracy**
[12]	2003	SLCT	Frequency Pattern Mining	Offline	Open Source	No	0.637
[60]	2003	Splunk	-	Online	Commercial	Yes	-
[54]	2008	AEL	Heuristics	Offline	No Open Source	Yes	0.754
[53]	2009	LKE	Clustering	Offline	No Open Source	No	0.563
[61]	2009	Loggly	-	Online	Commercial	Yes	-
[51]	2010	LFA	Frequent Pattern Mining	Offline	No Open Source	No	0.652
[17]	2013	SHISO	Clustering	Online	No Open Source	No	0.669
[52]	2015	LogCluster	Frequent Pattern Mining	Offline	Open Source	Yes	0.665
[55]	2016	LenMa	Clustering	Online	Open Source	No	0.721
[13]	2016	LogMine	Clustering	Offline	No Open Source	Yes	0.694
[62]	2016	Spell	Longest Common Subsequence	Online	Paid	No	0.751
[57]	2017	Drain	Parsing Tree	Online	Open Source	No	0.865
[39]	2018	POP	Iterative partitioning	Online	Open Source	No	0.986
[56]	2020	LogParse	Word Classification	Online	Open Source	No	0.978
[59]	2020	Craftsman	prefix-tree and frequent patterns	Online	No Open Source	No	0.962
[14]	2021	OILog	keyword extraction	Online	No Open Source	No	0.961

Splunk [60] and Loggly [61] are the commercial log analysis tools included with automated log parsers. These tools are mainly used for enterprise on-premises or software as a service (SaaS).

3) LOG ANALYSIS

The log analysis makes the log more readable and understandable. The clear and simplified outlook of the logs assists in problem detection and troubleshooting. With log analysis, one can extract patterns and knowledge which could guide and facilitate IT infrastructure monitoring, problem diagnosis, root cause analysis, and troubleshooting. After carefully studying the automated log analysis tools, we compared the tools based on techniques used, merit criteria such as product type, mode, availability, and industrial utility, as shown in Table 8. LogAider tool [63] works on the spatial and temporal correlation mining between events to extract fatal events effectively. Tool show 95% similarities in the analysis as compared to the report generated by the admin. LogLens [64] works on the concept of finding the relationship between the typical workflow execution log sequence and streaming logs to find anomalies. This technique speeds

TABLE 8. Log analysis tools studied in the systematic literature review.

Ref	Year	Tool Name	Technique	Merit Criteria			
				Product Type	Mode	Availability	Industrial Utility
[63]	2018	LogAider	Temporal-Correlation	On-Premises	Online	Open-Source	No
[65]	2019	Priolog	Temporal Analysis	SaaS	Online	No Open-Source	No
[66]	2012	Graylog	-	On-Premises	Online	Open-Source	Yes
[67]	2010	ELK	-	On-Premises/ SaaS	Online/ offline	Open-Source	Yes
[68]	2011	Fluentd	-	SaaS	Online	Open-Source	Yes
[64]	2018	LogLens	ML	-	-	Commercial	Yes
[69]	2010	Sumo Logic	-	SaaS	Online	Open-Source / Commercial	Yes
[70]	2014	Logz.io	-	SaaS	Online	Open-Source / Commercial	Yes

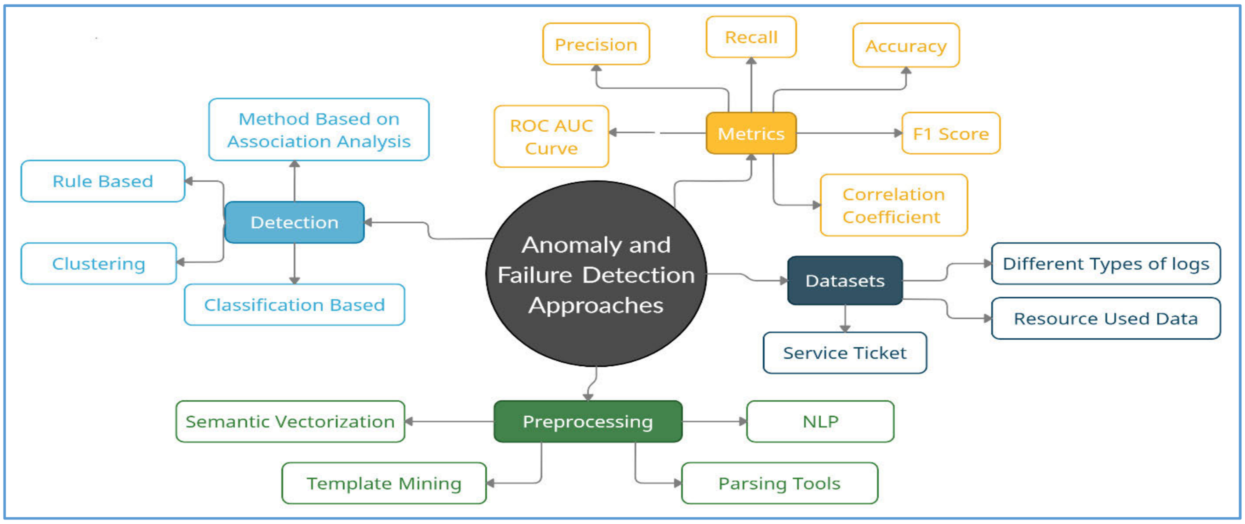


FIGURE 10. Anomaly and failure detection approaches studied in systematic literature review.

up the problem detection and saves up to 12096x person-hours. Priolog [65] utilizes temporal analysis and prioritization techniques to split the enormous log data into small groups. This grouping helps to identify problems fast and root cause analysis. According to the systematic literature review, various researchers designed and implemented many log analysis tools, providing adequate results.

Along with tools present in the literature, a few enterprise log analysis tools are also available. Graylog [66], Elastic Stack [67], and Fluentd [68] are a few of the open-source tools used by many companies to analyze logs and monitor the infrastructure. Sumo Logic [69] and Logz.io [70] is cloud-based data analytics tool that helps to analyze log data quickly and support system monitoring and troubleshooting problems in real-time.

D. ANOMALY AND FAILURE DETECTION APPROACHES

The abnormal pattern does not correspond to the expected behaviors recorded as an anomaly in the system. This strange behavior propagates and may be responsible for failures. This section elaborates on a systematic literature review of anomaly and failure detection approaches in the existing literature. Fig. 10 is outlined after a critical analysis of scholarly publications.

Fig. 10 briefs about the detection approaches, pre-processing or feature extraction techniques, datasets utilized in the study, and evaluation metrics applied to check the performance of models. In the existing literature, great work has been done on various infrastructures by utilizing different types of logs (Syslog [71], event log [72], switch log [73], exception log [74], RAS log [49], etc.), IT service ticket data [75] and resource used data [76] as a dataset for analysis. Traditional machine learning (ML) algorithms generally execute on extracted features. Thus, pre-processing or feature extraction of logs is obligatory. Template mining [77], sematic vectorization [78], [79], use of NLP techniques [80] are the popular approaches applied for pre-processing or feature extraction of the selected dataset. After a rigorous analysis of scholarly articles, we can say predominantly detection approaches are classified into four categories, such as rule-based approach [11], a method based on association analysis [81], Clustering [82], and classification-based methods [83], [84]. Different evaluation metrics are used to measure the performance of these algorithms. Most frequently used evaluation metrics are precision [85], recall [86], accuracy [87], F1 score [88], correlation coefficient [89].

The studied research components are listed collectively in Table 9 with specific fascinating properties. Expressly, properties like particular infrastructure, dataset, technical approaches for detection, pre-processing or feature extraction techniques, stated performance, and relevant insight indicated to particularize the strengths and weaknesses of the study in Table 9. The approaches are divided into subcategories based on detection strategy, namely anomaly detection, failure detection, fault detection, impactful service detection, and run-time problem detection.

A significant study has been done in this domain. However, current systems demand correspondence within the alerts and events to weaken the untrue warnings [90]. According to Studiawan and Sohel [80], imbalance situations in log data can be the reason for the low performance of the anomaly detectors.

This paper predominantly emphasizes general types of anomalies and failure conditions that occur due to the software systems' spontaneous flaws and results in downtime. External causes of the failure, such as cyber-attacks and malicious activities, are out of scope for this systematic literature review, although they best fit system security.

1) THE RULE-BASED APPROACH

The rule-based approach compares logs against a set of expert-defined rules to identify the abnormal behavior of the software system. This approach primarily engages graph models for the early detection of anomalies or failures.

Jia *et al.* [11] introduces time-weighted control flow graphs (TCFG) to catch the normal execution of cloud system. An anomaly alarm is embossed when abnormal behavior is observed in the transactional and operational log of the Hadoop system. Nandi *et al.* [91] employed a control flow graph (CFG) technique to overcome the need for instrumentation requirements or application specification assumptions. Model claim approximately 90% recall rate for sequential and distributed anomaly detection in OpenStack. Jia *et al.* [77] claim an average of 90% precision and 80% recall with a hybrid graph model. This model runs in two layers. First layer work on the calculation of service topology based on the frequency of the log. Graph-based mining takes place to design time-weighted control flow graphs (TCFGs) for anomaly detection.

2) CORRELATION AND ASSOCIATION-BASED APPROACH

Farshchi *et al.* [72] proposed a regression-based approach to encounter anomalies in the execution of amazon DevOps operations. Correlation between the operation log and the resource used data was established to check operations activities' effect on cloud resources. B *et al.* [89] proposed a LADT (lightweight anomaly detection tool) to detect anomalies in virtual machines on the cloud. An anomaly detected alarm raises when the correlation coefficient value drops below the threshold level. The correlation coefficient value is calculated using node-level data and VM level metrics. Di *et al.* utilized various types of data to find the correlation and detect anomalous behavior. Data sources include reliability, availability, and serviceability (RAS) log; job scheduling log; the log regarding each job's physical execution tasks; and the I/O behavior log used for joint analysis [49]. Di *et al.* recorded meantime to interruption (MTTI) as 3.5 days for the whole Mira system during the experiment. Nie *et al.* [86] identified pair-wise relationships in sequences to form the clusters using a multivariate relationship graph. An anomaly is recorded in the physical plant sensor's dataset if one or more pairwise relationships are breached.

3) CLUSTERING

In a clustering-based approach, a cluster of logs is generated depending upon the similarity of features. The size of the log message, recorded timestamp, and the log level are a few of the parameters applied for clustering. Log entries similar to each other are combined in the same cluster and dissimilar in others. Cluster with very few log instances likely to be anomalous. To assist the developers by detecting a problem with the help of forming the clusters of event log sequences. The LogCluster [52] algorithm designed by Lin *et al.* He *et al.* [92] proposed Log3C, a unique cascading clustering algorithm-based framework to detect impactful system problems by accessing log event sequence and KPIs (Key performance indicators). This framework forms clusters of massive data promptly and precisely by iteratively sampling, clustering, and matching log sequences. CRUDE (Combining Resource Usage Data and Error) [93] employed console log resource used data of Ranger supercomputer for accurate error detection. Jobs with abnormal resource usage were identified with the help of making clusters of similar behavior nodes. Du and Cao [82] observed the relation between log sequences and corresponding behavior patterns to point out Hadoop and LANL data anomalies. In the study Chen *et al.* [94], the hierarchical clustering algorithm was used to form clusters to identify anomalies based on their score, but they neglected the incompleteness of logs. Recently, Yang *et al.* proposed a novel reclustering algorithm by improving K-means to detect a BlueGene/L and Thunderbird system fault. The distributed Memory model of Paragraph Vectors (PV-DM) utilized to procure low-dimensional log vectors then an improved K-mean algorithm was applied to form the clusters [95].

4) CLASSIFICATION BASED APPROACH

In the research of IT infrastructure monitoring, ML and DL techniques were utilized to classify log data and detect anomalies and failures in the system. Most of the literature

TABLE 9. Anomaly and failure detection approaches, techniques, metrics, and performance in it infrastructure studied in systematic literature review.

Ref	Year	Strategy	Infrastructure	Dataset	Technical Approach	Preprocessing/ Feature Extraction	Techniques/ Algorithm Used	Performance	Relevant Insights
[82]	2015		Hadoop & LANL	System logs	Clustering	Erroneous behavior	Hierarchical clustering	F-Score Improved by 13%	Log puerility ignored while computing recent log sequence anomaly score
[93]	2016		Ranger Supercomputer	Console log & Resources used data	Clustering	Mutual Information (I) and Entropy (H)	Hierarchical clustering, PCA	True positive rate: 80%	Event sequence and resource usage data were utilized to find a probable sequence that may cause failure
[91]	2016		Spark & Hadoop cluster	Execution log	Control Flow Graph	Template Mining	Multi-modal signal of text and temporal vicinity, OASIS	-	Healthy execution flow seize via CFG, variations results in an anomaly
[89]	2016		Cloud System	Log	Correlation	-	LADT algorithm	Correlation coefficient value below the threshold	Add on CPU and I/O intensive; latency sensitivity characteristics will improve correlation performance
[71]	2017		virtual network	System log	ML	word2vec algorithm	Random Forest, MLP, Gaussian NB	Accuracy: ≈ 90%	Other causes need to explore supplement to stress behavior
[11]	2017		IBM public Cloud	Trasactional, Operational log	Control flow graph	Template mining	Time-weighted control flow graph	Precision, Recall: 80%	The Black-box approach applied to detect anomalous behavior
[77]	2017		IBM Cloud	Execution log	Control flow graph	Template mining	Time-weighted control flow graph	Average Precision:90% Recall: 80%	Prior system wisdom not needed to design TCFG
[16]	2017		HDFS & Open Stack	System log	DL	Log Parsing	LSTM	TP: 100%, FP: 38.2% to 1.1% for 10% data	100% detection accuracy archived by virtue of user feedback
[96]	2018		HDFS	System log	DL	Log Parsing, logkey2vec	CNN	Precision: 95%, Recall:95 F1-measure: 96	CNN furnish better results as compared to LSTM and MLP. No application-specific details required
[81]	2018	**Anomaly Detection**	HPC	System log	DL	Word2vec, TF-IDF	LSTM	Accuracy, Precision, Recall, F1-score: 0.99	Model accomplish the best result but fail to consider the unexpected change in log
[72]	2018		Amazon Web Services	Operational Event log, Resources Matrix	Correlation/ regression analysis	NA	Multiple regression models	-	Operation behavior & change in resource state exploited to detect injected faults, fail to detect coeval faults
[78]	2019		HDFS	System log	DL	Semantic Vectorization (TF-IDF)	Attention-based Bi-LSTM	Precision: 0.92, Recall: 0.97, F1-Score: 0.95 with 20% injection ratio	Capable to detect and deal with unstable log events and sequences
[97]	2019		Aerospace server	Time series data	Stochastic recurrent neural network		Deep Bayesian network	Precision: 0.77, Recall: 0.95, F1- Score: 0.85	Works powerfully for different devices, which generates time-series data
[85]	2019		Yahoo & NAB	Time series data	DL	time series	CNN	Precision: ~1, Recall: between 0.001 − 0.36.	Able to detect wide range of deviations in time series data
[79]	2019		HDFS & BLG	System log	DL	template2vec	LSTM	Precision: 0.95 Recall: 0.94 F1 Score: 94	Statistical analysis causes false alarm, use of semantic information
[88]	2019		HPC	System log	ML		Autoencoder	F1 Score:> 0.99 Accuracy: 90%	Fabrication of labeled & unbiased log data is highly complex
[98]	2019		Network	Network Log	ML	NA	Conditional Varia-tional Autoencoder	Precision : 91.5%, Recall: 74.0%	CVAE gives improved results without preprocessing & feature extraction
[73]	2019		Network	Switch log	ML	TF-IDF	Positive-unlabeled Learning SVM	F1 score: 99.51%, Macro-F1: 95.32%, Micro-F1: 99.74%	Able to elicit top-n vital words for every anomaly group
[86]	2020		Sensor & Hard Disk	Sensor log, Hard Disk data	Graph Theory	Neural machine translation	Multivariate relationship graph,	Recall: 58%	Feature engineering depends on the domain knowledge
[80]	2020		Operating System	System log	ML, DL	Drain, TF-IDF, GloVe	LSTM	Precision Recall: 99%	Imbalance data managed by utilizing the class balancing method
[83]	2020		HDFS & Oil industry	System log	ML	Parsing	Confidence-guided parameter adjustment method	Precision :98.2%, Recall 95.2% F-measure : 96.7%	Generalized solution for all types of systems, Worked together on linking and training of dataset
[99]	2020		IoT system	Event log	ML	Drain, Word2Vec	Random Forests, Naive Bayes, and Neural Networks	Precision: 94.7%, Recall: 94.0% F-measure: 0.94	Vectorization of log data reduces the computational cost
[64]	2020		Data Center	Trace log	ML	Exemplary stateless algorithm, exemplary stateful algorithm	Apriori based technique	4074.31% improvement in Storage Server, 1629.41% PCAP	Instead of parsing system learns the structure from the correct log at runtime
[8]	2021		Sensors	System log	DL	Correlation matrix enables	RNN–LSTM	F1 Score: 96%, 92%, 97% for log data, multivariate data, HR datasets respectively	Need to cultivate a more extensive dataset along with more activation functions
[76]	2011		HPC	System log and Resource used data	Correlation	Template Generation	Event correlation analysis in space and time	-	Able to provide nodes, correlated events & date of event sequences about failure
[100]	2016		cloud system	System log	Correlation	Redundant filtering, meaningless words	DBSCAN algorithm	Precision: 98%	Fault keyword matrix technique enhances the log classification accuracy
[87]	2018		OpenStack & Hadoop	System log and VM operation data	-	Workflow extraction process, Mapping workflow to tasks	Building automata from labeled log sequences,	Accuracy: >95%,	Do not act on new failure conditions, also not able to differentiate between same workflows
[101]	2019	**Failure Detection**	HPC	Failure logs and Re-source use data	Correlation	-	Fisher's z-score, Bonferroni correction, Time-bin Extraction	z-scores range from 3.75 to 12.13	Spikes in resource utilization may be the cause of failure
[74]	2019		OpenStack	Exception log	ML, NLP	word2vec	KNN, Naive Bayes, NN, and RF	Accuracy: 96.45%, F1-score: 99.084%	Approach possibly to apply for failure detection of internal periodic tasks
[49]	2019		IBM Blue Gene/Q Mira	RAS log, Job Scheduling/ Cobalt job log, Task Execution log, and I/O log	Correlation	-	-	3.5 days of MTTI for the whole Mira System	The approach can detect the locality features that influence the job execution
[102]	2012		HPC	System log	ML	Abnormality based filtering	Naive classifier, New classifier	Accuracy: 85%	Abnormality based screening upgrade accuracy of location & time detection
[84]	2019	**Fault Detection**	Network	Router log	ML/DL	Semantic Analysis	LSTM	-	Need to focus on multi-dimensional warning information
[95]	2021		BlueGene/L	System log	Clustering	PV-DM language model	K-means, Reclustering	Accuracy: 98%	Need to focus on semantic similarity threshold based on log characteristics
[92]	2018	**Impactful Service Problem Detection**	Cloud system	System log+ KPIs	Clustering and Correlation	Parsing using a regular expression, Vectorization using IDF weighting	Hierarchical Agglomerative Clustering (HAC), Log3C	Precision: 0.877 Recall: 0.883	Need to check the validity of approach on other cloud-based infrastructures
[103]	2018	**Run Time Problem**	Cloud System	System log	ML	POP Parser, N- gram Features and SVM	OCSVM- semi supervised	0.823 Precision, 0.843 recall, 0.33 F - measure	Superior on the unbalanced training dataset

focuses on the classification approach for detection using ML and DL techniques, as reflected in Table 9.

- Machine Learning-Based Techniques:

In the research of IT infrastructure monitoring, ML techniques are utilized to classify log data and detect anomalies and failures in the system. As reflected in Table 9, most literature focuses on the classification approach for detection using ML techniques. Various algorithms like decision trees, random forest, SVM, Naïve Bayes, and Gaussian NB are applied to classify log data. Few researchers employed feature reduction, feature selection, and feature extraction processes to improve the performance of classifiers. Word2Vec, a bag of words (BoW), Term Frequency - Inverse Document Frequency (TF-IDF), template2Vec are the literature's most prevailing feature extraction techniques. The classes are formed based on standard and anomalous behavior where outlier detected as an anomaly or failure Bertero *et al.* [71] extracted features by applying the Word2Vec approach on the system log followed by the Binary classifier, random forest, Gaussian NB to detect stress behavior of the virtual network. An unsupervised learning approach was utilized to detect anomalies in online streaming data [96] and privacy-aware abnormalities in the HPC system [97]. Also, Bronevetsky *et al.* [102] introduced the unsupervised model to enumerate individual node abnormality. To analyze the reason for network anomaly, Wang *et al.* [84] exploited Isolation Forest, OneClassSVM, and LocalOutlierFacto unsupervised algorithms Yan *et al.* [104] Proposed a novel EKF-CS-D-ELM hybrid classification method to resolve the air handling unit's (AHU) fault detection and diagnosis issue. The authors applied a cost-sensitive dissimilar extreme learning machine. Authors claimed more accurate, fast and robust fault diagnosis results over support vector machine (SVM).

HitAnomaly [105] anomaly detection framework developed to apprehend semantic data in the template sequence and parameter value by applying encoder. LogTransfer [106] proposed a method to transfer the unusual observation of source software systems to target software systems by considering global word co-occurrence and local context information and tackling logs in different formats. Studiawan and Sohel [80] suggested using the class balancing method to deal with the challenge of handling imbalance in data.

- Deep Learning-Based Techniques:

DeepLog [16] transformed log records into natural language sequences by applying LSTM neural network model and claimed 100% anomaly detection accuracy. Wang *et al.* [81], Meng *et al.* [79] Processed log data using NLP techniques and generated vectors provided to LSTM for anomaly detection to mitigate the false alarms. At the same time, Zhang *et al.* [78] presented a sequence of semantic vectors to Bi-LSTM (Bidirectional Long Short-Term Memory) Borghesi *et al.* [88] implemented a semi-supervised autoencoder-based strategy to avoid trouble in data labelling. Xie *et al.* [83] applied a confidence-guided anomaly identification model by blending multiple algorithms to combat concept drift. Supervised models such as random forest, naïve Bayes, and neural networks outperform anomaly detection on vectorized data [99].

E. FAILURE PREVENTION APPROACHES

When any system swerves from its intended work and cannot accomplish system-required functions, this situation is called a failure condition. Even if we handle such failure conditions very promptly, it introduces downtime. Such unavailability in the continuously working large-scale system is unexpected and dissatisfactory for users. The problem discovery in the components and connections in IT infrastructure is possible by observing the unusual behavior of the system. Although fault is determined, gathering the required information such as location, path, involved components, cause, etc., is extremely difficult. Thus, we need to build a system that can predict the failure condition in prior. Another way to prevent the failure condition is to find the root cause of the problem and take corrective actions to avoid it in the future. In IT infrastructure, the conventional procedure is to leverage the precious system logs to predict failure preemptively.

1) PREDICTION

This section will elaborate on a systematic literature review of anomaly and failure prediction approaches in the existing literature. Fig. 11 is outlined on critical analysis of scholarly publication. Fig. 11 briefs about the prediction approaches, feature extraction techniques, datasets utilized in the study, and evaluation metrics applied to check the performance of models. In the existing literature, significant work has been done on various infrastructures by utilizing benchmark datasets [104], released for experimentation purposes, and real-time datasets [75] developed in the specific environment. Traditional machine learning (ML) algorithms generally execute on extracted features. Thus, the extraction of logs is obligatory. Bag of words [107], Term Frequency - Inverse Document Frequency (TF-IDF) [75], Global Vectors for Word Representation (GloVe) [106], feature matrix algorithm [108] are the popular approaches applied for preprocessing or feature extraction of the selected dataset.

After a rigorous analysis of scholarly articles, we can say various types of predictions fabricated to avoid failure situations in different infrastructures by taking their kind into account. Table 10 indicates the different strategies for prediction such as failure prediction (supercomputer [109], VM [110], cloud system [111], event prediction (HPC [112], BlueGene/L [9], IoT [113]), fault prediction (Distributed system [114], job status prediction (cluster [115]), correct maintenance time prediction (ATM [108]), vending machine [116]), remaining useful time prediction (Hard disk [117]), incident prediction (Server [118]), Server crash prediction (VM [119]).

Predominantly prediction approaches are classified into three categories, such as Machine learning techniques [9], Deep learning techniques [109], [120], and Time-series techniques [110], [118]. Different evaluation metrics are used to measure the performance of these algorithms. Most

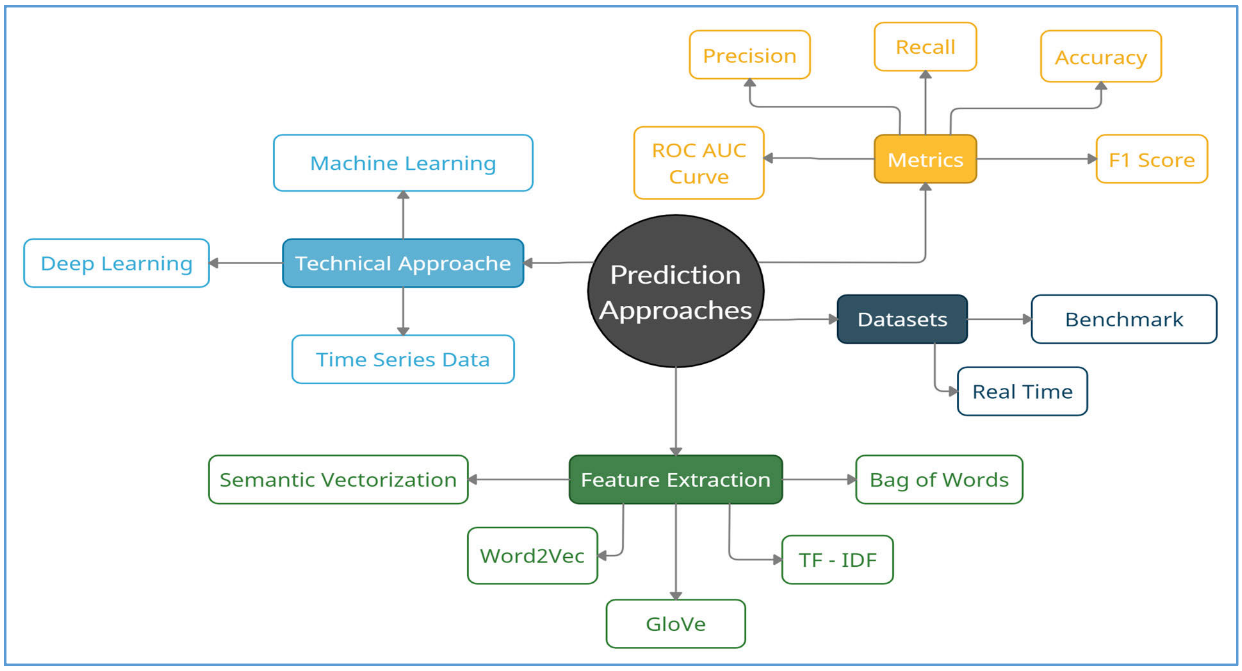


FIGURE 11. **Prediction approaches studied in systematic literature review.**

frequently used evaluation metrics are precision [121], recall [19], accuracy [10], F1 score [122], AUC [111] and lead time for prediction [109].

The studied research components are listed collectively in Table 10 with specific fascinating properties. Expressly, properties like particular infrastructure, dataset, technical approaches for detection, pre-processing or feature extraction techniques, stated performance, and relevant insights indicated to particularize the strengths and weaknesses of the study in Table 10.

- Failure Prediction

Zheng *et al.* [7] affirmed betterment in fault tolerance (reduce service unit loss by up to 52.4%) by applying a genetic algorithm-based method. Seer [123] can predict 54% of the system's hardware failures. Karakurt *et al.* [124] utilized machine learning approaches to predict failure in the oracle database. In comparison, Rawat *et al.* [110] applied a time series stochastic model to predict VM failure in cloud infrastructure. Researches augmented the concept of TF-IDF with LSTM [120] and deep CNN algorithms [19] to predict the failure in HPC and Hadoop infrastructure, respectively. Doomsday [125] enforced time-based learning to detect the rare computer node failure and time-based phrases as prediction mechanisms Li *et al.* [111] proposed a framework that can predict node failure ultra-large cloud computing and helps DevOps (software development and IT operations) in establishing AIOps (Artificial Intelligence for IT Operations). Elsayed and Zulkernine projected PredictDeep [122] framework for cloud security anomaly detection and prediction by applying a combination of graph analytics and deep learning techniques. It also successfully reduced the false alarm rate of anomaly prediction

- Event Prediction

Researchers have explored probability, correlation, machine learning, and deep learning techniques in the existing literature for event prediction. According to Gainaru *et al.* [126], event prediction in the HPC system is vital to acquire proactive actions for failure identification, tolerance, and recovery Fu *et al.* [127] proposed a tool for a system administrator for semi-automated detection of the root cause failure event by applying a three-step approach.

- Fault prediction

Gainaru *et al.* [128] suggested a hybrid approach (signal analysis and data mining) for fault prediction in an HPC system. He also claimed that the hybrid approach outperformed than individual execution. Pal and Kumar [114] applied distributed log mining using ensemble learning (DLME) on network logs.

- Job Status Prediction

Saadatfar *et al.* [10] served the Bayesian network as a data mining technique to encounter the relationship between workload characteristics and job failures. The analyzed data assists in detecting the failure pattern in the auvergrid system. Yoo *et al.* [115] utilized machine learning classifiers for job status prediction by characterizing the patterns of task executions in a job with the classes of successful and unsuccessful job statuses. The authors applied 13 resource-usage-related fields measuring resource usages in the job logs and feed them as features to machine learning mechanisms

- Correct Maintenance Time Prediction

Predicting the correct maintenance time and scheduling maintenance action can relieve failure situations in any hardware system. ML techniques practiced for maintenance time

TABLE 10. Failure prediction approaches, techniques, metrics, and performance in IT infrastructure studied in systematic literature review.

Ref	Year	Strategy	Infrastructure	Dataset	Technical Approach	Preprocessing/ Feature Extraction	Techniques/ Algorithm Used	Metrics Used	Relevant Insights
[7]	2010	Failure Prediction	IBM Blue Gene/P	Job log, RAS log	Genetic Algorithm	Michigan encoding method	Pearson's correlation	-	Precision and recall decrease with a growing lead time
[121]	2013		HPC	System log	Signal Analysis and Data Mining	Signal Characterization, Correlation	-	Precision: 90%	Delivers poor results on noise and periodic signals
[120]	2016		Webserver cluster (WSC)& mailer server cluster (MSC)	Console log	DL	TF-IDF	LSTM	(WSC) Recall: 90.9% Interval: 73 Min Frequency:66.2% (MSC) Recall: 80%, Interval: 22 Min, Frequency: 30.4%	DL outperforms in the context of PR-AUC, predictable interval, and predictable frequency
[124]	2017		Oracle	Database log	ML	Normalization methods	Random Forest	Recall: 75.7% Precision: 84.9%	Time series based prediction & regression techniques may boost the performance
[110]	2018		Virtual Machine	Time Series data VM	ML	-	ARIMA, Box–Jenkin method, random indexing (RI) and SVM	RMSE: 0.0457 MAE: 0.0344 MASE: 0.603, Mean % error : 0.016	Useful for prompt fault tolerance & prototype private cloud
[19]	2019		CMRI-Hadoop and bluegene/L	System log	DL	TF-IDF	Deep CNN algorithms	Precision: 98.14% Recall: 98.08 F1Score: 98.11	DL techniques suitable to give good comprehension on Syslog without revealing business-sensitive information
[107]	2019		Network	Log messages, trouble ticket	ML	bag-of-words	SVM	-	Pattern-based approach outperforms over traditional keyword-based/ template based
[125]	2019		Cray System	Job log, ALPS log, Console log	ML	Time correlation, Data integration		Precision: 98% Recall: 83% Lead time: 2Min	The accomplishment of prediction counts on a selection of optimal learning window interval and derived lead time
[129]	2019		Hard Disk	SMART attributes	ML	-	iNNE, iForest and LOF	Recall decreases with increase in cross ration	Wrong selection of attributes may impact accuracy
[109]	2020		HPC	System log	DL	Tokenization	LSTM	Recall: 86% 88%, Precision:88% Accuracy: 80% Lead time: 3 min	Compiler based approach brings forth unfamiliar areas of research for prediction
[111]	2020		Cloud system	Time Series data	ML	Temporal, Spatial and Build feature	LSTM and random forest	AUC: 0.9, 0.92 for LSTM and random forest	Augmentation of feature engineering & data sampling techniques obtain better results with low computational power
[122]	2020		Hadoop & cloud system	System log	DL	Graph model	LSTM	F1-measure:92%-94%, False alarm rate 0.03	Model training with contextual features demonstrates a rise in performance in the case of mean based GCN
[126]	2011	Event Prediction	Mercury BlueGene/L LANL	System log	Probability	Regular expressions, correlation chains	Probability	Precision: 85%	Time-efficient as event rules can be updated easily, fails to deliver sufficient precision
[112]	2012		HPC, Hadoop, BlueGene/L	System log	Correlation	n-ary sequence, apriori-like algorithms	Event correlation graphs (ECGs)	15.01 Min for Hadoop, 23.34 Min for HPC, 29.58 BlueGene/L	Use of filtering techniques improves prediction result
[9]	2014		BlueGene/L	System log	ML	Adaptive Semantic, Duplicate Removal Filter	Naive Bayes	Prediction Window: 0.553 Sec	Comprehensive accuracy decreases on the introduction of noise in data
[113]	2019		IOT	Bigdata log	DL	Log sequence	Hidden Markov Model (HMM), Autoregressive Integrated Moving Average Model	Root Mean Square Error (RMSE) reduced by 46.65%	Can't update dynamically for new failure conditions; a selection of optimal windows can improve accuracy
[128]	2012	Fault Prediction	HPC	Event log	Correlation	Hierarchical Event Log Organizer	Sequential GRITE algorithm	MTTF: 5 hours, Wasted Time: 20%, Recall: 50%	The hybrid approach on signal analysis and data mining outperforms than individual
[114]	2019		Distributed System	Network log	Ensemble Learning	Feature matrix	Weighted majority approach with a random forest	Accuracy: 84.5 Precision: 0.7, Recall: 0.71 F1-score: 0.71	Not working on dynamically updating network system
[10]	2012	Job Status Predictio	auvergrid	Activity Log	Data Mining	-	Bayesian Networks	96% accuracy	Prediction accuracy relies on features, training window size & log availability
[115]	2016		Genepool scientific cluster	Job log	ML	-	Random Forests	83.6% recall and 94.8% precision	Supports to minimize time, resource waste & cost due to failures
[108]	2017	Correct Maintenance Time Prediction	ATM	Error events and ticket events	ML	Feature ranking algorithm	XGBoost, Random Forest, and Ada Boost	0.82 AUC	The use of a feature selection helps to reduce the negative impact of excess features
[116]	2019		Venting Machine	Machine log, maintenance data, post-service reports	ML	Kruskal-Wallis test, Fishers Exact test, T-test	SVM, RF, and GBM (Gradient Boosting Machine)	80% accuracy	Working for limited data and fixed log conditions
[117]	2018	Remaining Useful Life Prediction	Hard Disk	SMART attributes	ML	Recursive Feature Elimination	Bayesian Network	-	The Bayesian network can be used for Disk failure prediction
[130]	2018		Hard Disk	SMART attributes	ML	Rank- sum test	Random forest	FDR: 97.67% FAR: 0.017% family "B", FDR: 100% FAR: 1.764% family ''S,'' FDR: 94.89% FAR: 0.44% family ''T''	SMART parameters are not sufficient to get all required details
[118]	2019	Incident Prediction	Netflix hulu	Incident data	ML	-	Hybrid model of NN,ARIMA, RF	MAPE: 25.67% RF: 26.55%, ARIMA: 27.01, NN: 32.41%	Unreported incidents not considered for prediction, need a generalized model for cloud system
[75]	2020		Online Banking Service System	Historical Alert data, insident ticket	ML	Topic model, TF-IDF	XGBoost	0.82 F1-score	The endowment of each feature and textual features are preponderant than statistical features
[119]	2020	Server Crash Predict	Virtual Machine	Hardware information, kernel status, and Syslog	ML	-	Random Forests	Minute level precision: 93.33% Hour level precision: 87.33%	Capturing and extracting informative hardware status serves to refine recall

TABLE 11. Required data, techniques, metrics and performance of root cause analysis in systematic literature review.

Ref	Year	Infrastructure	Required data for Analysis	Technique Used	Metrics Used	Performance
[131]	2017	Spark	Execution Log and Garbage Collection Log	Weighted Factor	Accuracy	Accuracy: 88.125%
[132]	2018	Cloud systems	Metrics Data of Services and Resource Utilization	Similarity Score	Mean average precision	15%-17% improvement in mean average precision
[133]	2019	OpenStack	Log Event Sequence and Cloud Service Behavior	Vectored Event Sequence	Macro-precision, macro-recall	Macro-precision: 97.08%, Macro-recall: 95.45%
[134]	2019	Cloud systems	Metrics and Event Logs	Event-Driven Activity Monitoring	Precision, Recall, F1-	Precision: 0.97 Recall: 0.31

prediction enforced on ATM [108] and vending machines [116] in the literature.

- Remaining Useful Life Prediction

To predict the health of the hard disk Self-Monitoring, Analysis and Reporting Technology (SMART) attribute of a hard disk provided to Bayesian Network [117] and Random Forest [130].

- Incident Prediction

Roumani and Nwankpa [118] used a hybrid model that engages ml and time series (arima) techniques to prophesy cloud incidents. Moreover, the ewarn [75] framework proposed to predict general incidences in online service systems by utilizing historical log data.

2) ROOT CAUSE ANALYSIS

Table 11 presented the required data, techniques, metrics, and performance of root cause analysis in a systematic literature review. Root cause analysis is the approach to define, understand and resolve the fault in the system. Root cause analysis is necessary to find the underlying cause of the problem to identify appropriate solutions. Furthermore, the primary reason can also pertain to the precise point in employing corrective action and preventing failure [135]. Lu *et al.* [131] designed a model to identify the root cause of application delay in the Spark system by utilizing weighted factors to determine the probability of root cause. CPU, memory, network, and disk are four components included to find the root cause of abnormalities. Weng *et al.* [132] developed a solution to assist cloud administrators in localizing the anomaly's root cause. This solution works effectively on VM and process level and encounters root cause even if anomaly happens due to multiple reasons. Weng *et al.* took advantage of both application layer and underlay infrastructure to discover the root cause. Graph base framework proposed by Brandón *et al.* [136] to find the root cause analysis for service-oriented and micro service architectures.

The authors also claimed that graph base methods outperformed by 19.41% over the machine learning approach Yuan *et al.* [133] applied a learning-based approach in OpenStack cloud service to track the root cause for anomalies. The stated process learns log patterns from past experience and is used for knowledge building. According to Konno and Défago [134], root cause analysis is momentous to ensure the cloud system's quality of service (QoS). Experiments performed on time series monitoring data of injected faults and real-time strategy.

IV. THE ARCHITECTURE OF THE PROPOSED SYSTEM

The proposed architecture of failure prediction in IT Infrastructure to avoid failure conditions is shown in figure 12. The proposed methodology pipeline is divided into four phases: 1) Preprocess raw log data and extract valuable features for the Deep Learning Models. 2) Training model trains the Deep Learning models considering the provided features 3) Model testing investigate the effectiveness of the trained Deep Learning Model, and 4) Deliver output in the form of prediction along with supporting actions.

The first block shows any raw log data as a dataset available for experimentation purposes. The second block represents a log parsing step, which derives the log template from the raw log by using the log parsing tool. It is the process of converting unstructured logs to structured logs. Log parsing reduces the log data size by removing the redundant logs generated through the same logging statement. The third block depicts the feature extraction process, which derives the semantic vector sequence from the log template records. This semantic analysis will be performed to identify relevant features from massive log data with the help of Natural Language Processing techniques. By considering only relevant features, we will be able to avoid the challenges in handling massive log data. These extracted features will be put forward to the fourth block to train the model. A deep learning model will be trained to detect the probable failures and identify the failure pattern by analyzing historical data.

The fifth block illustrates the process of model testing. In this phase, the testing dataset (balanced dataset) will be supplied to the trained model. The time window is introduced to get sufficient lead time for a prediction. Late predictions as less time before the failure would be of no use as system admin would not have time to take mitigation actions. To deal with this essential parameter, we use log data in a specific time window. Sufficient lead time of failure prediction will be helpful to take corrective actions and avoid downtime.

The last part of the architecture shows the activities to be performed after getting the model results. An alert will be generated to notify the system admin of the prediction of any potential failure.

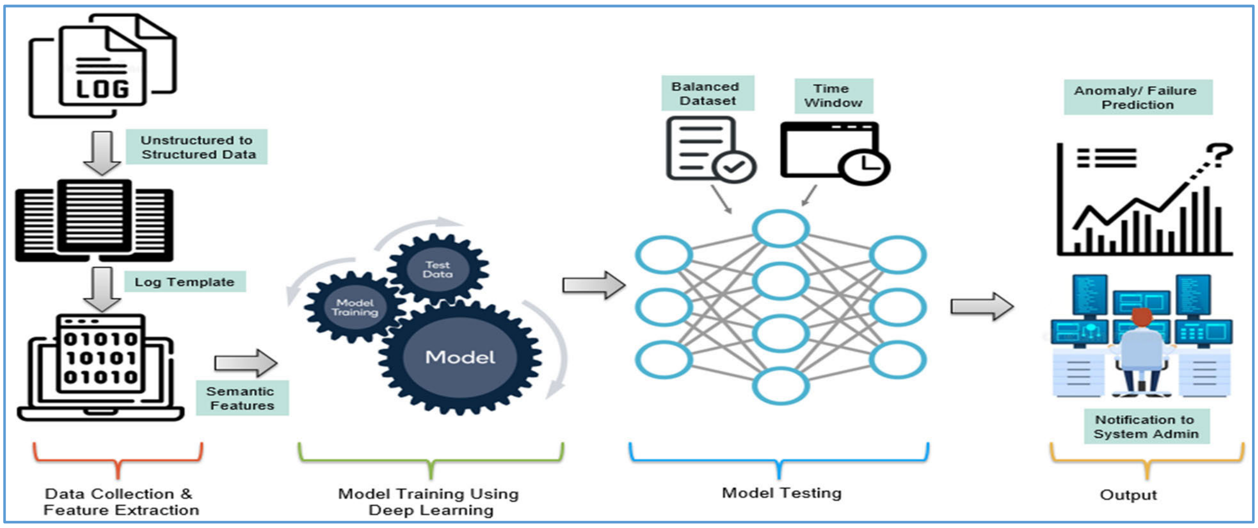

Unstructured to Structured Data
Log Template
Semantic Features
Test Data
Model Training
Model
Balanced Dataset
Time Window
Anomaly/ Failure Prediction
Notification to System Admin
Data Collection & Feature Extraction
Model Training Using Deep Learning
Model Testing
Output

FIGURE 12. **The architecture of proposed system for IT infrastructure failure prediction.**

V. EXPERIMENTS

The authors performed experimentation to fulfil the proposed architecture's first phase (data collection and feature extraction). All the required datasets and parsing tools, and feature extraction approaches utilized for the experiment are shortlisted on rigorous analysis of existing literature. In the literature review, more focus was given on studying the availability of log datasets, tools and techniques applied for preprocessing, detection and prediction operations etc.

In a way, we can say that selection of parsing tools and vectorization techniques for experimentation is the output of this systematic literature review. Similarly, other aspirants can benefit from this SLR to identify the appropriate tools, techniques, or approaches while working in the IT infrastructure monitoring domain.

In IT infrastructure failure detection and prediction first and foremost action is to collect the log data from selected infrastructure. The gathered log is always present in raw format. Such log data cannot be served directly for the detection or prediction process. Thus, it is obligatory to metamorphose unstructured raw log data into the structured log. The processed structure logs are undertaken for subsequent analysis. From the proposed architecture, log parsing followed by sematic analysis for feature extraction is targeted for implementation. This section emphasis on the experimentation modules: dataset, log parsing, and semantic analysis.

A. DATASET

In accordance with the conducted literature review, various datasets are utilized in the study and released for further experimentation, as shown in Table 6. We have picked up one sample dataset from each category for experimental activities. Logs of various infrastructures were collected for experimentations, such as HDFS from distributed system category, BGL HPC from the supercomputer category, Linux from the operating system category, Android from the mobile system category, Apache from the server application category, and Proxifier from the software category.

B. LOG PARSING

Every single log message is inscribed by a logging statement that records the state of the system execution. Log messages registered with log header and message contents. Log header is an amalgamation of id, state, timestamp, level, etc. Moreover, message contents are a combination of the constant and variable parts. The developer wrote the constant string as a printing statement and variable component updates on execution and permeated current state particulars. The constant string imparts the log template of the log message and stays intact for the entire event presence. The primary aspiration of log parsing is to alter every log message into a particular template. Fig. 13 exhibits the elements of the sample HPC system log. HPC raw log message included different log header parameters (LogId, Node, Component, State, Time, and Flag) and message contents (Content). Furthermore, contents conveyed to procreate a unique event template.

Many automated parsers are open-source and grant accuracy adequately concerning the investigation done and rendered in Table 7. "Drain" parser transforms logs into the most anticipated format. Also, the environment set up for the tool's execution is not much complicated and easy to configure with confine system configuration. All selected log entries are parsed by executing the "Drain" automated parser.

Table 12 illustrates the compendious of obtained results on the execution of Drain parser on different types of dataset. From derived results as stated in Table 12, we can observe that "Drain" provides acceptable accuracy for all types of infrastructures.

C. SEMANTIC ANALYSIS

Most ML and DL models for detection or prediction are not prepared to work directly on normal text data. As a result,

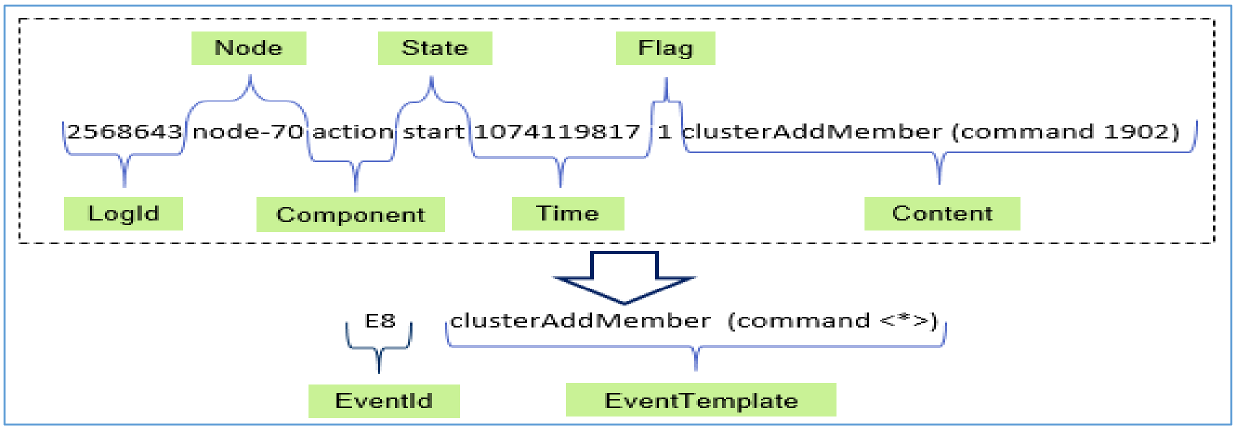

Node
State
Flag
2568643 node-70 action start 1074119817 1 clusterAddMember (command 1902)
LogId
Component
Time
Content
E8
clusterAddMember (command <*>)
EventId
EventTemplate

FIGURE 13. Elements of sample HPC system log.

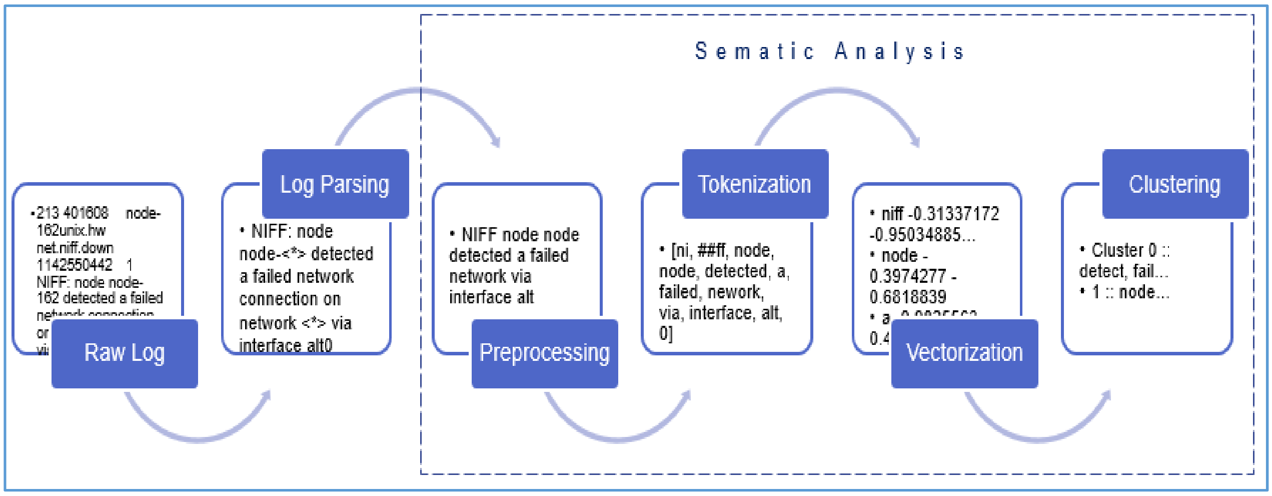

Semantic Analysis
•213 401608 node-162unix.hw net.niff.down 1142550442 1 NIFF: node node-162 detected a failed network connection
Raw Log
Log Parsing
• NIFF: node node-<*> detected a failed network connection on network <*> via interface alt0
• NIFF node node detected a failed network via interface alt
Preprocessing
Tokenization
• [ni, ##ff, node, node, detected, a, failed, nework, via, interface, alt, 0]
• niff -0.31337172 -0.95034885...
• node - 0.3974277 - 0.6818839
Vectorization
Clustering
• Cluster 0 :: detect, fail...
• 1 :: node...

FIGURE 14. Process of semantic analysis and sample log conversion received on execution of step.

TABLE 12. Experimental results of drain parser on various datasets.

Infrastructure Category	Infrastructure	Size of Data	Number of log Messages	Number of Unique Template	Template Max Length	Parsing Accuracy
Distributed System	HDFS	1.47 GB	11,175,629	30	29	0.99
Supercomputer	BGL HPC	708.76 MB	4,747,963	619	376	0.99
Operating System	Linux	2.25 MB	25,567	488	134	0.96
Mobile System	Android	3.38 GB	30,348,042	76,923	188	0.91
Server	Apache	4.90 MB	56,481	44	42	1
Application Software	Proxifier	2.42 MB	21,329	9	27	0.87

feature extraction or a digital delineation of the event template is obligatory. We have performed semantic analysis by squeezing the event template's sematic knowledge and transforming each event template into vectors. This vectorization positively facilitates preventing the influence of change in the syntax of logs. Our semantic analysis experimentation was achieved with the aid of the BERT (Bidirectional Encoder Representations from Transformers) model.

Fig. 14 exemplifies the process of semantic analysis. For semantic analysis, the event template (for example, NIFF: node node-$< * >$ detected a failed network connection on network $< * >$ via interface alt0) undergoes the following steps: pre-processing, tokenization, vectorization, and clustering.

- We begin by removing all non-character emblems from the event template, such as special symbols, punctuation marks, numbers, operators, etc. For example: ‘‘NIFF node node detected a failed network via interface alt0’’
- Tokenization is the technique of partitioning a string into a list of tokens. We have performed tokenization by applying ‘‘BertTokenizer’’ of the ‘‘BERT pre-train model’’. For example : ‘‘[ni, ##ff, node, node, detected, a, failed, nework, via, interface, alt, 0]’’
- Then data in pertinent format is forwarded to the pre-train model for word embedding. Finally, the vectors acquired for each token of the event template. For example: niff −0.31337172 −0.95034885. . ., node −0.3974277 −0.6818839, a −0.9825563 -0.4542647
- In the last step, the clusters are formed based on the semantic of the vectors. Here we used the simple k nearest neighbor approach (KNN) to form the clusters of similar meaning words. For example: Cluster 0 :: clusteraddmember Cluster 1 :: command

VI. DISCUSSION

This section conveys a panorama of noteworthy points from the systematic literature review on IT infrastructure monitoring. The analysis targeted to furnish the answers to research questions and satisfy objectives as stated in Table 1.

- RQ1: *How are log entries valuable for troubleshooting the failure?*

Various IT infrastructure components generate different types of log data on the execution of events. The system logs are rich in information and provide all the details about the activities executed on the IT infrastructure components. System logs are considered as the primary source of data as it records the software’s runtime information. Thus, recorded logs in the IT infrastructure are a valuable resource to track the issues in the system and handle it correctly. By processing the log, one can obtain the details about the timestamp, log level, log message, resources involved, etc. this data helps identify and analyze the problem.

Information that arises after processing massive log data can monitor the system’s behavior; it examines the root causes of the issues. Also, historical logs are helpful to understand the behavior of the system and identify the failure pattern. The analysis of logs (sequence of records) is advantageous to gather the details about the execution of activities and resources utilized. This data requires troubleshooting the identified problems in the system. Considering the properties of system logs and data generation on processing on them add great worth in maintaining the health of IT infrastructure by troubleshooting the failure.

- RQ2: *What are different IT Infrastructures and logs used to perform research experiments?*

Various infrastructures are considered to monitor and handle the failure conditions in the studied literature. Researchers have utilized different types of infrastructures such as distributed systems, supercomputers, operating systems, mobile systems, server applications, standalone software, etc., during the researcher to monitor the system’s health and detect or predict failure. Also, they have applied various techniques to exploit the different types of logs and other metrics to gather the correct information for troubleshooting. Majorly, Syslogs are employed for analysis in the existing literature. We observed that most researchers have functioned on supercomputers such as HPC, BLG, and IBM Blue Gene. A substantial volume of research is made in Hadoop and HDFS, accompanied by cloud systems like OpenStack, IBM Public Cloud, and the Webserver. Enough research has been done in the pinpointing and avoidance of defeats in the network. Handful scholars have concentrated on the hardware system to forecast the maintenance epoch and its strength. Identification of node failure in a virtual machine, IoT, belongs to infrastructures probed by infrequent scholars. Finally, yet importantly, research has already been undertaken on software applications. The failure in software applications can cause computer system downtime.

- RQ3: *What is the performance of different approaches for anomaly and failure detection in IT Infrastructure?*

In the systematic literature review, we have studied different approaches used for preprocessing, anomaly & failure detection, and prevention, as discussed in section III and represented in Figures 9, 10, and 11. After rigorous analysis of all these approaches, techniques, and results, we have listed a few popular and efficient methods for different operations. 1) Preprocessing: natural language processing (NLP) for preprocessing logs as logs combine text and numbers and log message plays a vital role in analyzing problems. Thus, rather than statistical analysis, the semantic analysis provides better results. The systematic literature review reveals that semantic scrutiny is preferable over statistical analysis to infer the relevant meaning from log data. Thus, many researchers have applied NLP techniques for preprocessing log data. Also, efficient feature extraction supports improving detection and prediction accuracy. 2) Anamoly or failure detection: classification using machine learning or deep learning techniques provides better accuracy than rule-based or method-based approaches. Also, the presentation of logs in the form of time series data is one of the ways the researcher explores to claim better results. In addition, a handful of researchers have explored autoencoder semi-supervised learning approaches. The exploitation of an autoencoder is advantageous on the chance of big unlabeled log data.

- RQ4: *What are the different existing techniques available to prevent and predict failure in IT Infrastructure monitoring?*

Failure prevention is possible by heterogeneous ways such as maintaining the health of components, finding the root cause of the failure, avoiding known causes, calculating the remaining useful time, monitoring the behavior, predicting the failure condition, etc. Different predictions have been made in the existing literature, such as failure propagation path, failure or fault or event prediction, or the accurate time for maintenance. Additionally, systems are enforced to forecast the maintenance period, remaining valuable life of the hard disk, and stress in the network to maintain the system's health. Thus, primarily, failure prevention is possible by predicting the failure situation with sufficient lead time. For the prediction using massive log data, sophisticated deep learning approaches imparts improved performance. Many researchers powerfully used Recurrent Neural Network, Convolutional Neural Network, LSTM, Bi-LSTM, etc. Considering the massive amount of logs, researchers recently preferred deep learning approaches to train the models. With the help of advanced, sophisticated deep learning techniques, it is possible to design a system that can update dynamically and improve the accuracy of failure prediction and prediction lead time.

- RQ5: *What are the different state-of-the-art tools and techniques used for log monitoring and analysis?*

In the SLR, we evaluated various automated tools for log preprocessing (Table 7) and log analysis (Table 8) based on the technique applied and four merit criteria: mode, availability, industry utility, and accuracy. Many parsing tools are available with adequate accuracy; thus, upcoming researchers can use any tool in accordance with their requirements instead of developing the new. Many commercial tools are accessible in open source and payment mode to visualize the analysis of logs. The simplified and clear view of log analysis certainly helps in troubleshooting the problem. However, current prediction tools or frameworks have many limitations such as lack of accuracy, resulting from certain assumptions, insufficient lead time, etc. Thus there is a demand for virtuous prediction tools which can apprise failure states with adequate lead time.

- RQ 6: *What are the distinguished limitations of existing literature?*

We did an extensive literature review on existing research and highlighted potential research gaps. Significant research has been done on different IT infrastructures using various types of log data, but the proposed solutions are system-specific. Limitations of the existing literature are discussed below:

i. Existing models in the literature are system-specific:

There is no solution available that can be applicable for all types of infrastructure. Different Infrastructures are obtainable and provide various features based on the utilization of components log generated in a different format. The above stated is the main reason for the system-dependent solutions.

ii. The logs considered with an assumption:

The system's log is the primary source of information that delivers details about the execution of events and component utilizations. Sometimes, logging instructions are not appropriately written; thus, logs do not produce the required information. Research has been conducted on such log data assuming that the generated log is complete and accurate.

iii. Preprocessing may result in loss of essential data:

The preprocessing carried out on log data by executing abstraction, filtering, encoding, removing unimportant data, etc., may lead to loss of critical information. This loss in essential data may decrease the accuracy of anomalies or failure detection and prediction. Also, the removal of some data will convert logs into incomplete records.

iv. Only significant anomalies/ failures can be detected:

More focus is given only on the detection of substantial anomalies or failures. Effective means the anomalies or failures occurs frequently and cause significant losses. Therefore, existing models cannot detect every anomaly or failure in the selected infrastructure.

v. The current system does not provide information for taking necessary actions:

Available models can detect or identify failure but do not provide information like the cause of failure, location or path, components involved which can help to adopt necessary measures. The additional details about failure will be helpful to take quick action and avoid propagation of failure and reduce downtime in the system.

vi. Not sufficient prediction lead time:

The estimated forecasting time in the existing anomalies or failure prediction system is inadequate to grab remedial actions. The researchers are designing a predictive system that can notify failure in advance, but the correctness of prediction declines with rising lead time.

vii. Systems not updating dynamically:

Existing systems are not picking up dynamically, which cannot detect or predict anomalies or failures that have never appeared in history or are unreported. But, likely, new irregularities or failure conditions will not occur in the upcoming future.

viii. Concurrent anomalies/ failures cannot be detected:

Furthermore, the existing model cannot detect or predict anomalies or failures that co-occur. Hence there is a need for a system that can handle such issues.

ix. Human intervention required:

Human intervention is essential in an earlier unobserved log sequence; consequently, no fully automatic system exists. Also, system administrators will need to handle the failure situation and take corrective actions. Human intervention introduces human errors due to human limitations with his respective knowledge, availability, and qualities.

x. Root cause analysis present only for past failure:

Also, Root cause analysis is available only for past failures. As a result, new failure conditions cannot be handled quickly, leading to increased downtime and associated losses.

VII. FUTURE DIRECTIONS

This coherent literature review was conducted on the scholarly publications extracted from Scopus, IEEE, ACM, and

Web of Science databases until early 2021. The relevant publications are limited by selected keywords applied while searching in the database. The manual screening was conducted on available full-text articles to finalize the list of publications for detailed analysis; thus, it is not assured that all the articles from the literature are studied thoroughly. In the systematic literature review, more focus was given on the tools and techniques used to handle failure conditions in IT infrastructures using log data. For these reasons, the evaluation may endure the threat ofinclination. The paper presents the proposed system architecture for IT infrastructure failure detection and prediction as a further solution to existing literature options. This systematic literature review focused more on the following points: (1) IT infrastructures used for study in the literature, (2) log data to detect anomaly and failure conditions, (3) activities needed to handle failure conditions and (4) various publicly available automated tools for log parsing or log analysis.

The proposed methodology discussed in section IV is under research and evaluation. From the proposed architecture, log parsing followed by sematic analysis for feature extraction is targeted for implementation, and preliminary results are discussed in the paper.

- Design generalized solutions to detect or predict failure of any IT infrastructure

The existing IT infrastructure failure detection or prediction solutions are systems dependent due to the change in nature of components, connections, utility and log formats. Although significant research has been done in this area, the IT industry demands a generalized solution that will apply to any IT infrastructure. Thus, there is a need to design a generalized system that will monitor any IT infrastructure.

- Generate or collect required logs with enhanced quality

All the components in the IT infrastructure generate different types of logs, which are helpful to monitor and maintain the health of the system. But these logs are not available in a standard format and also have quality issues. To resolve this issue, identify and configure a tool that can gather the required logs from all infrastructure components. Also, remember the common features from different types of logs to improve data quality for further processing.

- Validate and improve failure prediction further with the help of already predicted events.

Data of previously predicted incidences can be further used and back feed to the system to validate further and improve the prediction model. Thus, a confidence-based system can be established to validate and improve prediction. In addition, the solution can be further strengthened to handle failure conditions proactively.

- Identify failure patterns based on historical data with a minimal set of log data.

Identify different failure patterns to improve prediction with minimal data sets as the system predicts more and more failure conditions. In some instances, the researcher may not have all debug level data or all level log data in this situation; a model trained with minimal data set will be effective. Such improvement in the technique can help to predict and prevent failure with the help of minimal logs.

- Create a monitoring console to show potential failure, performed actions, and different matrices.

User Interface (UI)-based monitoring consoles can be built to monitor the components in IT infrastructure better. This console can help to visualize the system appropriately. This console can provide a detailed view of the overall system. It may show what action and suggestions are provided and how many measures are taken manually or automatically. It can also have an idea of the system's confidence, prediction, and success rate. This console will also help to reduce the human intervention in IT infrastructure monitoring.

- The researcher can further design a remediation system with the help of Orchestrator Application Programming Interfaces (APIs).

An automated system can suggest corrective action on the identified anomalies and failure conditions. One can explore how these suggestions and techniques can be utilized and executed to avoid failures. The stated recommendations may be orchestrated with API to run automatically. These workflows will help to prevent failure conditions proactively. This solution will be helpful to update the system dynamically as per the runtime requirements.

VIII. CONCLUSION

The recent past has witnessed the flourish in the utilization of IT infrastructures. Extensive importance has been given to system logs to establish stable and reliable infrastructure. Many researchers have furnished immense efforts for efficient and compelling log analysis to detect and control failure conditions to evade downtime. This systematic literature review mainly probes the five main stages in the IT infrastructure monitoring framework: availability of the log data, log parsing, log analysis, anomaly or failure detection, and prevention techniques. Furthermore, we elaborated on the open-source as well as commercial automated tool kits used in IT infrastructure monitoring. On rigorous analysis of studied literature, we have derived ten prominent research gaps. In accordance with the exploration of these recent advances, we suggested novel insights and listed various future directions.

As a result of a systematic literature review, experimentation is performed with shortlisted parsing tools and feature extraction approaches. For experiments, the authors utilized the datasets from various infrastructures as suggested in Table 6. Also, a "Drain" open-source parser was applied to convert unstructured log to structured log, which gives acceptable accuracy for all infrastructures. BERT pre-train model was selected for semantic analysis based on the comparative study of feature extraction techniques in the available literature.

This systematic literature review and performed experimentation enable the forthcoming researchers to step into this encouraging and pragmatic field and empower them to fill their understanding gaps.

GLOSSARY

- Accuracy: Evaluate the accurately predicted samples. Accuracy can be calculated by: Accuracy = (TP + TN)/ (TP + FP + TN + FN) FN [137].
- ALPS - Application-Level Placement Scheduler
- API - Application Programming Interface
- ARIMA - Autoregressive Integrated Moving Average Model
- ATM - Automated Teller Machine
- AUC - Area Under the Curve
- BERT–Bidirectional Encoder Representations from Transformers
- BGL - Blue Gene/L
- Bi-LSTM - Bidirectional Long Short-Term Memory
- CFG - Control Flow Graph
- CNN - Convolutional Neural Network
- CPU - Central Processing Unit
- DBSCAN - Density-Based Spatial Clustering of Applications with Noise
- DevOps - Software Development and IT Operations
- DL–Deep Learning
- EKF-CS-D-ELM - Extended Kalman Filter - Cost-Sensitive Dissimilar Extreme Learning Machine
- F1-measure / F1-Score: The harmonic mean is derived by combining both the precision and recall values. F1-measure can be calculated as: $F1 - score = 2*$ (Precision $*$ Recall)/Precision + Recall [138], [139].
- FAR - False Alarm Rate
- FDR - Failure Detection Rates
- FN - False Negative
- FP - False Positive
- GBM - Gradient Boosting Machine
- GCN - Graph Convolutional Network
- GloVe - Global Vectors For Word Representation
- HDFS - Hadoop Distributed File System
- HPC - High-Performance Cluster
- HR - Heart Rate
- IoT - Internet of things
- I/O – Input / Output
- IT – Information Technology
- ITSM - IT Service Management
- KNN - k-Nearest Neighbor
- KPI - Key Performance Indicator
- LADT – Light-weighted Anomaly Detection Tool
- LANL- Los Alamos National Laboratory
- LSTM–Long Short-Term Memory
- LOF - Local Outlier Factor
- MAE - Mean Absolute Error
- MASE - Mean Absolute Scaled Error
- Macro-F1: Used to calculate the F1- score in the case of multi-class settings. Macro-F1 is also called a macro-averaged F1-score and calculated as simple arithmetic mean of F1 scores of each class. Reference [140]

$$Macro\ Average\ Precision = \frac{\sum_{k=1}^{k} \text{Precision}\kappa}{k}$$

$$Macro\ Average\ Recall = \frac{\sum_{k=1}^{k} \text{Recall}\kappa}{k}$$

$$Macro\ F1Score = 2 * \left(\frac{Macro\ Average\ Precision * Macro\ Average\ Recall}{Macro\ Average\ Precision^{-1} - Macro\ Average\ Recall^{-1}}\right)$$

- Micro-F1: Used to calculate the F1- score in the case of multi-class settings. Micro-F1 is also called a micro-averaged F1-score and is calculated by combining micro average precision and micro average recall [140].

$$Micro\ Average\ Precision = \frac{\sum_{\mathbf{k=1}}^{\mathbf{k}} \mathbf{TP}\boldsymbol{\kappa}}{\sum_{\mathbf{k=1}}^{\mathbf{k}} \mathbf{TotalColumns}}$$

$$Micro\ Average\ Recall = \frac{\sum_{\mathbf{k=1}}^{\mathbf{k}} \mathbf{TP}\boldsymbol{\kappa}}{\sum_{\mathbf{k=1}}^{\mathbf{k}} \mathbf{Total\ Rows}}$$

$$Micro\ F1Score = \frac{\sum_{\mathbf{k=1}}^{\mathbf{k}} \mathbf{TP}\boldsymbol{\kappa}}{\sum_{\mathbf{k=1}}^{\mathbf{k}} \mathbf{Total(Columns/Rows)}}$$

- ML– Machine Learning
- MLP - Multilayer Perceptron
- MTTF - Mean Time to Failure
- MTTI – Mean Time to Interruption
- NLP - Natural Language Processing
- NN - Neural network
- PCA–Principal Component Analysis
- Precision: Precision gives the number of correct predicted results divided by the number of predictions derived from the classifier [137]. Precision can be calculated by: Precision = TP/(TP + FP)
- RAS - Reliability, Availability, and Serviceability
- Recall: Recall provides the number of correct predicted results divided by a number of all applicable instances. It can be calculated by: Recall = TP/ (TP + FN) [137].
- RMSE - Root Mean Square Error
- RNN - Recurrent Neural Network
- ROC-AUC Curve: Receiver Operating Characteristic (ROC) is a two-dimensional

representation of the trade-off among the TP and FP rates [138]. This curve was utilized to calculate and compare the performance of the classifiers. Area Under Curve (AUC) is mainly applied for the binary classifiers equivalent to the concept of probability [141].

- RQ- Research Question
- SaaS - Software as a Service
- SMART - Self-Monitoring, Analysis and Reporting Technology
- SLR–Systematic Literature Reviews
- SVM–Support Vector Machine
- TCFG - Time-Weighted Control Flow Graphs
- TF-IDF–Term Frequency-Inverse Document Frequency

- TN - True Negative
- TP–True Positive
- UI – User Interface
- VM - Virtual Machine

REFERENCES

[1] A. Das, F. Mueller, C. Siegel, and A. Vishnu, "Desh: Deep learning for system health prediction of lead times to failure in HPC," in *Proc. 27th Int. Symp. High-Perform. Parallel Distrib. Comput.*, Jun. 2018, pp. 40–51.

[2] *Summary of the Amazon S3 Service Disruption in the Northern Virginia (US-EAST-1) Region*. Accessed: Sep. 18, 2021. [Online]. Available: https://aws.amazon.com/message/41926/

[3] H. Mi, H. Wang, Y. Zhou, M. R. T. Lyu, and H. Cai, "Toward fine-grained, unsupervised, scalable performance diagnosis for production cloud computing systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 6, pp. 1245–1255, Jun. 2013.

[4] W. Xu, L. Huang, A. Fox, D. Patterson, and M. I. Jordan, "Detecting large-scale system problems by mining console logs," in *Proc. ACM SIGOPS 22nd Symp. Operating Syst. Princ. (SOSP)*, 2009, pp. 37–44.

[5] D. A. Bhanage, "DigitalCommons university of Nebraska—Lincoln review and analysis of failure detection and prevention techniques in IT infrastructure monitoring," Library Philosophy Pract., Digit. Commons, Univ. Nebraska-Lincoln, Lincoln, NE, USA, Tech. Rep. 5248, 2021.

[6] Y. Tan and X. Gu, "On predictability of system anomalies in real world," in *Proc. IEEE Int. Symp. Model., Anal. Simul. Comput. Telecommun. Syst.*, Aug. 2010, pp. 133–140.

[7] Z. Zheng, Z. Lan, R. Gupta, S. Coghlan, and P. Beckman, "A practical failure prediction with location and lead time for blue gene/P," in *Proc. Int. Conf. Dependable Syst. Netw. Workshops (DSN-W)*, Jun. 2010, pp. 15–22.

[8] E. Elbasani and J.-D. Kim, "LLAD: Life-log anomaly detection based on recurrent neural network LSTM," *J. Healthcare Eng.*, vol. 2021, pp. 1–7, Feb. 2021.

[9] T. Pitakrat, J. Grunert, O. Kabierschke, F. Keller, and A. van Hoorn, "A framework for system event classification and prediction by means of machine learning," in *Proc. 8th Int. Conf. Perform. Eval. Methodologies Tools*, 2015, pp. 173–180.

[10] H. Saadatfar, H. Fadishei, and H. Deldari, "Predicting job failures in AuverGrid based on workload log analysis," *New Gener. Comput.*, vol. 30, no. 1, pp. 73–94, Jan. 2012.

[11] T. Jia, L. Yang, P. Chen, Y. Li, F. Meng, and J. Xu, "LogSed: Anomaly diagnosis through mining time-weighted control flow graph in logs," in *Proc. IEEE 10th Int. Conf. Cloud Comput. (CLOUD)*, Jun. 2017, pp. 447–455.

[12] R. Vaarandi, "A data clustering algorithm for mining patterns from event logs," in *Proc. 3rd IEEE Workshop IP Oper. Manag. (IPOM)*, Oct. 2003, pp. 119–126.

[13] H. Hamooni, B. Debnath, J. Xu, H. Zhang, G. Jiang, and A. Mueen, "LogMine: Fast pattern recognition for log analytics," in *Proc. 25th ACM Int. Conf. Inf. Knowl. Manage.*, May 2016, pp. 1573–1582.

[14] X. Duan, S. Ying, H. Cheng, W. Yuan, and X. Yin, "OILog: An online incremental log keyword extraction approach based on MDP-LSTM neural network," *Inf. Syst.*, vol. 95, Jan. 2021, Art. no. 101618.

[15] J. Sillito and E. Kutomi, "Failures and fixes: A study of software system incident response," in *Proc. IEEE Int. Conf. Softw. Maintenance Evol. (ICSME)*, Sep. 2020, pp. 185–195.

[16] M. Du, F. Li, G. Zheng, and V. Srikumar, "DeepLog: Anomaly detection and diagnosis from system logs through deep learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2017, pp. 1285–1298.

[17] M. Mizutani, "Incremental mining of system log format," in *Proc. IEEE Int. Conf. Services Comput.*, Jun. 2013, pp. 595–602.

[18] S. Jain, I. Singh, A. Chandra, Z.-L. Zhang, and G. Bronevetsky, "Extracting the textual and temporal structure of supercomputing logs," in *Proc. Int. Conf. High Perform. Comput. (HiPC)*, Dec. 2009, pp. 254–263.

[19] R. Ren, J. Cheng, Y. Yin, J. Zhan, L. Wang, J. Li, and C. Luo, "Deep convolutional neural networks for log event classification on distributed cluster systems," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2018, pp. 1639–1646.

[20] *Gartner Says the Future of IT Infrastructure is Always on, Always Available, Everywhere*. Accessed: Jun. 26, 2021. [Online]. Available: https://www.gartner.com/en/newsroom/press-releases/2018-12-03-gartner-says-the-future-of-it-infrastructure-is-always-on-always-available-everywhere

[21] *Gartner Announces Gartner IT Infrastructure, Operations & Cloud Strategies Conference 2020*. Accessed: Jun. 27, 2021. [Online]. Available: https://www.gartner.com/en/newsroom/press-releases/2020-11-02-gartner-announces-gartner-it-infrastructure-operations-and-cloud-strategies-conference-2020

[22] D. El-Masri, F. Petrillo, Y.-G. Guéhéneuc, A. Hamou-Lhadj, and A. Bouziane, "A systematic literature review on automated log abstraction techniques," *Inf. Softw. Technol.*, vol. 122, Jun. 2020, Art. no. 106276.

[23] M. Landauer, F. Skopik, M. Wurzenberger, and A. Rauber, "System log clustering approaches for cyber security applications: A survey," *Comput. Secur.*, vol. 92, May 2020, Art. no. 101739.

[24] R. B. Yadav, P. S. Kumar, and S. V. Dhavale, "A survey on log anomaly detection using deep learning," in *Proc. 8th Int. Conf. Rel., Infocom Technol. Optim. (Trends Future Directions) (ICRITO)*, Jun. 2020, pp. 1215–1220.

[25] D. Das *et al.*, "Failure prediction by utilizing log analysis: A systematic mapping study," in *Proc. ACM Int. Conf. Ser.*, Oct. 2020, pp. 188–195.

[26] S. He, P. He, Z. Chen, T. Yang, Y. Su, and M. R. Lyu, "A survey on automated log analysis for reliability engineering," 2020, *arXiv:2009.07237*.

[27] B. Kitchenham, O. P. Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, "Systematic literature reviews in software engineering—A systematic literature review," *Inf. Softw. Technol.*, vol. 51, no. 1, pp. 7–15, 2009.

[28] D. A. Bhanage and A. V. Pawar, "Bibliometric survey of IT infrastructure management to avoid failure conditions," *Inf. Discovery Del.*, vol. 49, no. 1, pp. 45–56, Feb. 2021.

[29] C. Wohlin, "Guidelines for snowballing in systematic literature studies and a replication in software engineering," in *Proc. 18th Int. Conf. Eval. Assessment Softw. Eng. (EASE)*, 2014, pp. 1–10.

[30] J. Zhu, S. He, J. Liu, P. He, Q. Xie, Z. Zheng, and M. R. Lyu, "Tools and benchmarks for automated log parsing," in *Proc. IEEE/ACM 41st Int. Conf. Softw. Eng., Softw. Eng. Pract. (ICSE-SEIP)*, May 2019, pp. 121–130.

[31] *GitHub—Logpai/Loghub: A Large Collection of System Log Datasets for AI-Powered Log Analytics*. Accessed: Jun. 27, 2021. [Online]. Available: https://github.com/logpai/loghub

[32] *The Computer Failure Data Repository (CFDR) | USENIX*. Accessed: Jun. 27, 2021. [Online]. Available: https://www.usenix.org/cfdr

[33] *SEC.gov | EDGAR Log File Data Set*. Accessed: Jun. 27, 2021. [Online]. Available: https://www.sec.gov/dera/data/edgar-log-file-data-set.html

[34] D. Cotroneo, L. De Simone, P. Liguori, R. Natella, and N. Bidokhti, "How bad can a bug get? An empirical analysis of software failures in the OpenStack cloud computing platform," in *Proc. 27th ACM Joint Meeting Eur. Softw. Eng. Conf. Symp. Found. Softw. Eng.*, Aug. 2019, pp. 200–211.

[35] *Failure Dataset*. Accessed: Jun. 27, 2021. [Online]. Available: https://figshare.com/articles/dataset/Failure_dataset/7732268/2

[36] *SecRepo—Security Data Samples Repository*. Accessed: Jun. 27, 2021. [Online]. Available: https://www.secrepo.com/

[37] A. Agrawal, R. Karlupia, and R. Gupta, "Logan: A distributed online log parser," in *Proc. IEEE 35th Int. Conf. Data Eng. (ICDE)*, Apr. 2019, pp. 1946–1951.

[38] S. Huang, Y. Liu, C. Fung, R. He, Y. Zhao, H. Yang, and Z. Luan, "Paddy: An event log parsing approach using dynamic dictionary," in *Proc. IEEE/IFIP Netw. Operations Manage. Symp. (NOMS)*, Apr. 2020, pp. 1–8.

[39] P. He, J. Zhu, S. He, J. Li, and M. R. Lyu, "Towards automated log parsing for large-scale log data analysis," *IEEE Trans. Dependable Secure Comput.*, vol. 15, no. 6, pp. 931–944, Nov. 2018.

[40] S. He, J. Zhu, P. He, and M. R. Lyu, "Loghub: A large collection of system log datasets towards automated log analytics," 2020, *arXiv:2008.06448*.

[41] B. Schroeder and G. Gibson, "A large-scale study of failures in high-performance computing systems," *IEEE Trans. Dependable Secur. Comput.*, vol. 7, no. 4, pp. 337–350, Oct. 2010.

[42] B. Schroeder and G. A. Gibson, "Disk failures in the real world: What does an MTTF of 1,000,000 hours mean to you?" in *Proc. 5th USENIX Conf. File Storage Technol.*, 2007, pp. 1–11.

[43] A. Oliner and J. Stearley, "What supercomputers say: A study of five system logs," in *Proc. 37th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN)*, Jun. 2007, pp. 575–584.

[44] J. Stearley and A. J. Oliner, "Bad words: Finding faults in Spirit's syslogs," in *Proc. 8th IEEE Int. Symp. Cluster Comput. Grid (CCGRID)*, May 2008, pp. 765–770.

[45] A. J. Oliner, A. Aiken, and J. Stearley, "Alert detection in system logs," in *Proc. 8th IEEE Int. Conf. Data Mining*, Dec. 2008, pp. 959–964.

[46] Z. Zheng, L. Yu, W. Tang, Z. Lan, R. Gupta, N. Desai, S. Coghlan, and D. Buettner, "Co-analysis of RAS log and job log on blue gene/P," in *Proc. IEEE Int. Parallel Distrib. Process. Symp.*, May 2011, pp. 840–851.

[47] K. Yan, J. Huang, W. Shen, and Z. Ji, "Unsupervised learning for fault detection and diagnosis of air handling units," *Energy Buildings*, vol. 210, Mar. 2020, Art. no. 109689.

[48] M. Hassani, W. Shang, E. Shihab, and N. Tsantalis, "Studying and detecting log-related issues," *Empirical Softw. Eng.*, vol. 23, no. 6, pp. 3248–3280, Dec. 2018.

[49] S. Di, H. Guo, E. Pershey, M. Snir, and F. Cappello, "Characterizing and understanding HPC job failures over the 2K-day life of IBM BlueGene/Q system," in *Proc. 49th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN)*, Jun. 2019, pp. 473–484.

[50] Z. Liu, T. Qin, X. Guan, H. Jiang, and C. Wang, "An integrated method for anomaly detection from massive system logs," *IEEE Access*, vol. 6, pp. 30602–30611, 2018.

[51] M. Nagappan and M. A. Vouk, "Abstracting log lines to log event types for mining software system logs," in *Proc. 7th IEEE Work. Conf. Mining Softw. Repositories (MSR)*, May 2010, pp. 114–117.

[52] R. Vaarandi and M. Pihelgas, "LogCluster—A data clustering and pattern mining algorithm for event logs," in *Proc. 11st Int. Conf. Netw. Service Manage. (CNSM)*, Nov. 2015, pp. 1–7.

[53] Q. Fu, J.-G. Lou, Y. Wang, and J. Li, "Execution anomaly detection in distributed systems through unstructured log analysis," in *Proc. 9th IEEE Int. Conf. Data Mining*, Dec. 2009, pp. 149–158.

[54] Z. M. Jiang, A. E. Hassan, P. Flora, and G. Hamann, "Abstracting execution logs to execution events for enterprise applications," in *Proc. 8th Int. Conf. Quality Softw.*, Aug. 2008, pp. 181–186.

[55] K. Shima, "Length matters: Clustering system log messages using length of words," 2016, *arXiv:1611.03213*.

[56] W. Meng, Y. Liu, F. Zaiter, S. Zhang, Y. Chen, Y. Zhang, Y. Zhu, E. Wang, R. Zhang, S. Tao, D. Yang, R. Zhou, and D. Pei, "Log-Parse: Making log parsing adaptive through word classification," in *Proc. 29th Int. Conf. Comput. Commun. Netw. (ICCCN)*, Aug. 2020, pp. 1–9.

[57] P. He, J. Zhu, Z. Zheng, and M. R. Lyu, "Drain: An online log parsing approach with fixed depth tree," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, Jun. 2017, pp. 33–40.

[58] M. Du and F. Li, "Spell: Online streaming parsing of large unstructured system logs," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 11, pp. 2213–2227, Nov. 2019.

[59] S. Zhang, L. Song, M. Zhang, Y. Liu, W. Meng, J. Bu, S. Yang, Y. Sun, D. Pei, J. Xu, and Y. Zhang, "Efficient and robust syslog parsing for network devices in datacenter networks," *IEEE Access*, vol. 8, pp. 30245–30261, 2020.

[60] *The Data-to-Everything Platform Built for the Cloud | Splunk*. Accessed: Apr. 6, 2021. [Online]. Available: https://www.splunk.com/

[61] *Log Analysis | Log Management by Loggly*. Accessed: Jun. 20, 2021. [Online]. Available: https://www.loggly.com/

[62] M. Du and F. Li, "Spell: Streaming parsing of system event logs," in *Proc. IEEE 16th Int. Conf. Data Mining (ICDM)*, Dec. 2016, pp. 859–864.

[63] S. Di, H. Guo, R. Gupta, E. R. Pershey, M. Snir, and F. Cappello, "Exploring properties and correlations of fatal events in a large-scale HPC system," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 2, pp. 361–374, Feb. 2019.

[64] B. Debnath, M. Solaimani, M. A. G. Gulzar, N. Arora, C. Lumezanu, J. Xu, B. Zong, H. Zhang, G. Jiang, and L. Khan, "LogLens: A real-time log analysis system," in *Proc. IEEE 38th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2018, pp. 1052–1062.

[65] B. Tak, S. Park, and P. Kudva, "Priolog: Mining important logs via temporal analysis and prioritization," *Sustainability*, vol. 11, no. 22, pp. 1–17, 2019.

[66] *Open*. Accessed: Apr. 6, 2021. [Online]. Available: https://www.graylog.org/products/open-source

[67] *ELK Stack: Elasticsearch, Logstash, Kibana | Elastic*. Accessed: Jun. 20, 2021. [Online]. Available: https://www.elastic.co/what-is/elk-stack

[68] *Fluentd | Open Source Data Collector | Unified Logging Layer*. Accessed: Sep. 9, 2020. [Online]. Available: https://www.fluentd.org/

[69] *Find answers hidden in your data | Sumo Logic*. [Online]. Available: https://www.sumologic.com/lp/brand/?utm_source=google&utm_medium=ppc&utm_campaign=APAC_Search_Branded_Top_Countries&utm_adgroup=sumo_log&utm_term=sumolog&gclid=CjwKCAjw6qqDBhB-EiwACBs6x8d0HbAiiOPrt4yW-10oHXJa-XGKQ1iIuckJBZqBfbUfzKJX1Xhs5RoCoxgQAvD_BwE Accessed: Apr. 6, 2021.

[70] *Logz.io: Cloud Observability for Engineers*. Accessed: Jun. 20, 2021. [Online]. Available: https://logz.io/

[71] C. Bertero, M. Roy, C. Sauvanaud, and G. Tredan, "Experience report: Log mining using natural language processing and application to anomaly detection," in *Proc. IEEE 28th Int. Symp. Softw. Rel. Eng. (ISSRE)*, Oct. 2017, pp. 351–360.

[72] M. Farshchi, J.-G. Schneider, I. Weber, and J. Grundy, "Metric selection and anomaly detection for cloud operations using log and metric correlation analysis," *J. Syst. Softw.*, vol. 137, pp. 531–549, Mar. 2018.

[73] W. Meng, Y. Liu, S. Zhang, D. Pei, H. Dong, L. Song, and X. Luo, "Device-agnostic log anomaly classification with partial labels," in *Proc. IEEE/ACM 26th Int. Symp. Quality Serv. (IWQoS)*, Jun. 2018, pp. 1–6.

[74] Y. Yuan, W. Shi, B. Liang, and B. Qin, "An approach to cloud execution failure diagnosis based on exception logs in OpenStack," in *Proc. IEEE 12nd Int. Conf. Cloud Comput. (CLOUD)*, Jul. 2019, pp. 124–131.

[75] N. Zhao, J. Chen, Z. Wang, X. Peng, G. Wang, Y. Wu, F. Zhou, Z. Feng, X. Nie, W. Zhang, K. Sui, and D. Pei, "Real-time incident prediction for online service systems," in *Proc. 28th ACM Joint Meeting Eur. Softw. Eng. Conf. Symp. Found. Softw. Eng.*, Nov. 2020, pp. 315–326.

[76] E. Chuah, G. Lee, W.-C. Tjhi, S.-H. Kuo, T. Hung, J. Hammond, T. Minyard, and J. C. Browne, "Establishing hypothesis for recurrent system failures from cluster log files," in *Proc. IEEE 9th Int. Conf. Dependable, Autonomic Secure Comput.*, Dec. 2011, pp. 15–22.

[77] T. Jia, P. Chen, L. Yang, Y. Li, F. Meng, and J. Xu, "An approach for anomaly diagnosis based on hybrid graph model with logs for distributed services," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, Jun. 2017, pp. 25–32.

[78] X. Zhang, Y. Xu, Q. Lin, B. Qiao, H. Zhang, Y. Dang, C. Xie, X. Yang, Q. Cheng, Z. Li, J. Chen, X. He, R. Yao, J.-G. Lou, M. Chintalapati, F. Shen, and D. Zhang, "Robust log-based anomaly detection on unstable log data," in *Proc. 27th ACM Joint Meeting Eur. Softw. Eng. Conf. Symp. Found. Softw. Eng.*, Aug. 2019, pp. 807–817.

[79] W. Meng, Y. Liu, Y. Zhu, S. Zhang, D. Pei, Y. Liu, Y. Chen, R. Zhang, S. Tao, P. Sun, and R. Zhou, "LogAnomaly: Unsupervised detection of sequential and quantitative anomalies in unstructured logs," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 4739–4745.

[80] H. Studiawan and F. Sohel, "Performance evaluation of anomaly detection in imbalanced system log data," in *Proc. 4th World Conf. Smart Trends Syst., Secur. Sustainability (WorldS)*, Jul. 2020, pp. 239–246.

[81] M. Wang, L. Xu, and L. Guo, "Anomaly detection of system logs based on natural language processing and deep learning," in *Proc. 4th Int. Conf. Frontiers Signal Process. (ICFSP)*, Sep. 2018, pp. 140–144.

[82] S. Du and J. Cao, "Behavioral anomaly detection approach based on log monitoring," in *Proc. Int. Conf. Behav., Econ. Socio-Cultural Comput. (BESC)*, Oct. 2015, pp. 188–194.

[83] X. Xie, Z. Jin, J. Wang, L. Yang, Y. Lu, and T. Li, "Confidence guided anomaly detection model for anti-concept drift in dynamic logs," *J. Netw. Comput. Appl.*, vol. 162, pp. 1–10, Feb. 2020.

[84] X. Wang, D. Wang, Y. Zhang, L. Jin, and M. Song, "Unsupervised learning for log data analysis based on behavior and attribute features," in *Proc. Int. Conf. Artif. Intell. Comput. Sci.*, Jul. 2019, pp. 510–518.

[85] M. Munir, S. A. Siddiqui, A. Dengel, and S. Ahmed, "DeepAnT: A deep learning approach for unsupervised anomaly detection in time series," *IEEE Access*, vol. 7, pp. 1991–2005, 2019.

[86] B. Nie, J. Xu, J. Alter, H. Chen, and E. Smirni, "Mining multivariate discrete event sequences for knowledge discovery and anomaly detection," in *Proc. 50th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN)*, Jun. 2020, pp. 552–563.

[87] P. Zhou, Y. Wang, Z. Li, G. Tyson, H. Guan, and G. Xie, "Logchain: Cloud workflow reconstruction & troubleshooting with unstructured logs," *Comput. Netw.*, vol. 175, Jul. 2020, Art. no. 107279.

[88] A. Borghesi, A. Bartolini, M. Lombardi, M. Milano, and L. Benini, "A semisupervised autoencoder-based approach for anomaly detection in high performance computing systems," *Eng. Appl. Artif. Intell.*, vol. 85, pp. 634–644, Oct. 2019.

[89] M. G. Jaatun, D. S. Cruzes, J. Angulo, and S. Fischer-Hübner, "Accountability through transparency for cloud customers," in *Proc. Int. Conf. Cloud Comput. Services Sci.*, vol. 1, 2016, pp. 38–57.

[90] D. C. Le and N. Zincir-Heywood, "A frontier: Dependable, reliable and secure machine learning for network/system management," *J. Netw. Syst. Manage.*, vol. 28, no. 4, pp. 827–849, Oct. 2020.

[91] A. Nandi, A. Mandal, S. Atreja, G. B. Dasgupta, and S. Bhattacharya, "Anomaly detection using program control flow graph mining from execution logs," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 215–224.

[92] S. He, Q. Lin, J.-G. Lou, H. Zhang, M. R. Lyu, and D. Zhang, "Identifying impactful service system problems via log analysis," in *Proc. 26th ACM Joint Meeting Eur. Softw. Eng. Conf. Symp. Found. Softw. Eng.*, Oct. 2018, pp. 60–70.

[93] N. Gurumdimma, A. Jhumka, M. Liakata, E. Chuah, and J. Browne, "CRUDE: Combining resource usage data and error logs for accurate error detection in large-scale distributed systems," in *Proc. IEEE Symp. Reliab. Distrib. Syst.*, Sep. 2016, pp. 51–60.

[94] C. Chen, N. Singh, and S. Yajnik, "Log analytics for dependable enterprise telephony," in *Proc. 9th Eur. Dependable Comput. Conf.*, May 2012, pp. 94–101.

[95] Z. Yang, S. Ying, B. Wang, Y. Li, B. Dong, J. Geng, and T. Zhang, "A system fault diagnosis method with a reclustering algorithm," *Scientific Program.*, vol. 2021, pp. 1–8, Mar. 2021.

[96] S. Lu, X. Wei, Y. Li, and L. Wang, "Detecting anomaly in big data system logs using convolutional neural network," *Proc. IEEE 16th Int. Conf. Dependable, Auton. Secur. Comput. IEEE 16th Int. Conf. Pervasive Intell. Comput. IEEE 4th Int. Conf. Big Data Intell. Comput.*, Aug. 2018, pp. 159–165.

[97] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, and D. Pei, "Robust anomaly detection for multivariate time series through stochastic recurrent neural network," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 2828–2837.

[98] K. Otomo, S. Kobayashi, K. Fukuda, and H. Esaki, "Latent variable based anomaly detection in network system logs," *IEICE Trans. Inf. Syst.*, vol. 102, no. 9, pp. 1644–1652, 2019.

[99] J. Wang, Y. Tang, S. He, C. Zhao, P. K. Sharma, O. Alfarraj, and A. Tolba, "LogEvent2vec: LogEvent-to-vector based anomaly detection for large-scale logs in Internet of Things," *Sensors*, vol. 20, no. 9, pp. 1–19, 2020.

[100] D.-Q. Zou, H. Qin, and H. Jin, "UiLog: Improving log-based fault diagnosis by log analysis," *J. Comput. Sci. Technol.*, vol. 31, no. 5, pp. 1038–1052, Sep. 2016.

[101] E. Chuah, A. Jhumka, S. Alt, D. Balouek-Thomert, J. C. Browne, and M. Parashar, "Towards comprehensive dependability-driven resource use and message log-analysis for HPC systems diagnosis," *J. Parallel Distrib. Comput.*, vol. 132, pp. 95–112, Oct. 2019.

[102] G. Bronevetsky, I. Laguna, B. R. de Supinski, and S. Bagchi, "Automatic fault characterization via abnormality-enhanced classification," in *Proc. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN)*, Jun. 2012.

[103] Y. Liu, J. Lv, S. Ma, and W. Yao, "The runtime system problem identification method based on log analysis," in *Proc. 27th Int. Conf. Comput. Commun. Netw. (ICCCN)*, Jul. 2018, pp. 1–7.

[104] K. Yan, Z. Ji, H. Lu, J. Huang, W. Shen, and Y. Xue, "Fast and accurate classification of time series data using extended ELM: Application in fault diagnosis of air handling units," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 49, no. 7, pp. 1349–1356, Jul. 2019.

[105] S. Huang, Y. Liu, C. Fung, R. He, Y. Zhao, H. Yang, and Z. Luan, "HitAnomaly: Hierarchical transformers for anomaly detection in system log," *IEEE Trans. Netw. Service Manage.*, vol. 17, no. 4, pp. 2064–2076, Dec. 2020.

[106] R. Chen, S. Zhang, D. Li, Y. Zhang, F. Guo, W. Meng, D. Pei, Y. Zhang, X. Chen, and Y. Liu, "LogTransfer: Cross-system log anomaly detection for software systems with transfer learning," in *Proc. IEEE 31st Int. Symp. Softw. Rel. Eng. (ISSRE)*, Oct. 2020, pp. 37–47.

[107] T. Kimura, A. Watanabe, T. Toyono, and K. Ishibashi, "Proactive failure detection learning generation patterns of large-scale network logs," *IEICE Trans. Commun.*, vol. 102, no. 2, pp. 306–316, 2019.

[108] J. Wang, C. Li, S. Han, S. Sarkar, and X. Zhou, "Predictive maintenance based on event-log analysis: A case study," *IBM J. Res. Dev.*, vol. 61, no. 1, pp. 121–132, 2017.

[109] A. Das, F. Mueller, and B. Rountree, "Aarohi: Making real-time node failure prediction feasible," in *Proc. IEEE 34th Int. Parallel Distrib. Process. Symp. (IPDPS)*, May 2020, pp. 1092–1101.

[110] A. Rawat, R. Sushil, A. Agarwal, and A. Sikander, "A new approach for VM failure prediction using stochastic model in cloud," *IETE J. Res.*, vol. 67, no. 2, pp. 1–8, Mar. 2018.

[111] Y. Li, Z. M. Jiang, H. Li, A. E. Hassan, C. He, R. Huang, Z. Zeng, M. Wang, and P. Chen, "Predicting node failures in an ultra-large-scale cloud computing platform," *ACM Trans. Softw. Eng. Methodol.*, vol. 29, no. 2, pp. 1–24, Apr. 2020.

[112] X. Fu, R. Ren, J. Zhan, W. Zhou, Z. Jia, and G. Lu, "Logmaster: Mining event correlations in logs of large-scale cluster systems," in *Proc. IEEE Symp. Reliab. Distrib. Syst.*, Oct. 2012, pp. 71–80.

[113] P. Wu, Z. Lu, Q. Zhou, Z. Lei, X. Li, M. Qiu, and P. C. K. Hung, "Bigdata logs analysis based on seq2seq networks for cognitive Internet of Things," *Future Gener. Comput. Syst.*, vol. 90, pp. 477–488, Jan. 2019.

[114] A. Pal and M. Kumar, "DLME: Distributed log mining using ensemble learning for fault prediction," *IEEE Syst. J.*, vol. 13, no. 4, pp. 3639–3650, Dec. 2019.

[115] W. Yoo, A. Sim, and K. Wu, "Machine learning based job status prediction in scientific clusters," in *Proc. SAI Comput. Conf. (SAI)*, Jul. 2016, pp. 44–53.

[116] S. Xiang, D. Huang, and X. Li, "A generalized predictive framework for data driven prognostics and diagnostics using machine logs," in *Proc. IEEE Region 10 Conf. (TENCON)*, Oct. 2018, pp. 695–700.

[117] I. C. Chaves, M. R. P. de Paula, L. G. M. Leite, J. P. P. Gomes, and J. C. Machado, "Hard disk drive failure prediction method based on a Bayesian network," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2018, pp. 1–7.

[118] Y. Roumani and J. K. Nwankpa, "An empirical study on predicting cloud incidents," *Int. J. Inf. Manage.*, vol. 47, pp. 131–139, Aug. 2019.

[119] X. Liu, Y. He, H. Liu, J. Zhang, B. Liu, X. Peng, J. Xu, J. Zhang, A. Zhou, P. Sun, K. Zhu, A. Nishi, D. Zhu, and K. Zhang, "Smart server crash prediction in cloud service data center," in *Proc. 19th IEEE Intersoc. Conf. Thermal Thermomech. Phenomena Electron. Syst. (ITherm)*, Jul. 2020, pp. 1350–1355.

[120] K. Zhang, J. Xu, M. R. Min, G. Jiang, K. Pelechrinis, and H. Zhang, "Automated IT system failure prediction: A deep learning approach," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2016, pp. 1291–1300.

[121] A. Gainaru, F. Cappello, M. Snir, and W. Kramer, "Failure prediction for HPC systems and applications: Current situation and open issues," *Int. J. High Perform. Comput. Appl.*, vol. 27, no. 3, pp. 273–282, Aug. 2013.

[122] M. A. Elsayed and M. Zulkernine, "PredictDeep: Security analytics as a service for anomaly detection and prediction," *IEEE Access*, vol. 8, pp. 45184–45197, 2020.

[123] B. Ozcelik and C. Yilmaz, "Seer: A lightweight online failure prediction approach," *IEEE Trans. Softw. Eng.*, vol. 42, no. 1, pp. 26–46, Jan. 2016.

[124] I. Karakurt, S. Ozer, T. Ulusinan, and M. C. Ganiz, "A machine learning approach to database failure prediction," in *Proc. Int. Conf. Comput. Sci. Eng. (UBMK)*, Oct. 2017, pp. 1030–1035.

[125] A. Das, F. Mueller, P. Hargrove, E. Roman, and S. Baden, "Doomsday: Predicting which node will fail when on supercomputers," in *Proc. Int. Conf. High Perform. Comput., Netw., Storage Anal.*, Nov. 2018, pp. 108–121.

[126] A. Gainaru, F. Cappello, J. Fullop, S. Trausan-Matu, and W. Kramer, "Adaptive event prediction strategy with dynamic time window for large-scale HPC systems," in *Managing Large-scale Systems Via the Analysis of System Logs and the Application of Machine Learning Techniques*. Cascais, Portugal, Oct. 2011, pp. 1–8, doi: 10.1145/2038633.2038637.

[127] X. Fu, R. Ren, S. A. McKee, J. Zhan, and N. Sun, "Digging deeper into cluster system logs for failure prediction and root cause diagnosis," in *Proc. IEEE Int. Conf. Cluster Comput. (CLUSTER)*, Sep. 2014, pp. 103–112.

[128] A. Gainaru, F. Cappello, M. Snir, and W. Kramer, "Fault prediction under the microscope: A closer look into HPC systems," in *Proc. Int. Conf. High Perform. Comput., Netw., Storage Anal.*, Nov. 2012.

[129] X. Gao, S. Zha, X. Li, B. Yan, X. Jing, J. Li, and J. Xu, "Incremental prediction model of disk failures based on the density metric of edge samples," *IEEE Access*, vol. 7, pp. 114285–114296, 2019.

[130] J. Shen, J. Wan, S.-J. Lim, and L. Yu, "Random-forest-based failure prediction for hard disk drives," *Int. J. Distrib. Sensor Netw.*, vol. 14, no. 11, Nov. 2018, Art. no. 155014771880648.

[131] S. Lu, B. Rao, X. Wei, B. Tak, L. Wang, and L. Wang, "Log-based abnormal task detection and root cause analysis for spark," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, Jun. 2017, pp. 389–396.

[132] J. Weng, J. H. Wang, J. Yang, and Y. Yang, "Root cause analysis of anomalies of multitier services in public clouds," *IEEE/ACM Trans. Netw.*, vol. 26, no. 4, pp. 1646–1659, Aug. 2018.

[133] Y. Yuan, H. Anu, W. Shi, B. Liang, and B. Qin, "Learning-based anomaly cause tracing with synthetic analysis of logs from multiple cloud service components," in *Proc. IEEE 43rd Annu. Comput. Softw. Appl. Conf. (COMPSAC)*, Jul. 2019, pp. 66–71.

[134] S. Konno and X. Defago, "Approximate QoS rule derivation based on root cause analysis for cloud computing," in *Proc. IEEE 24th Pacific Rim Int. Symp. Dependable Comput. (PRDC)*, Dec. 2019, pp. 33–42.

[135] *Root Cause Analysis (RCA) for IT—BMC Blogs*. Accessed: Apr. 30, 2020.[Online]. Available: https://www.bmc.com/blogs/root-cause-analysis/

[136] Á. Brandón, M. Solé, A. Huélamo, D. Solans, M. S. Pérez, and V. Muntés-Mulero, "Graph-based root cause analysis for service-oriented and microservice architectures," *J. Syst. Softw.*, vol. 159, Jan. 2020, Art. no. 110432.

[137] J. Tohka and M. van Gils, "Evaluation of machine learning algorithms for health and wellness applications: A tutorial," *Comput. Biol. Med.*, vol. 132, May 2021, Art. no. 104324.

[138] Y. Liu, Y. Zhou, S. Wen, and C. Tang, "A strategy on selecting performance metrics for classifier evaluation," *Int. J. Mobile Comput. Multimedia Commun.*, vol. 6, no. 4, pp. 20–35, Apr. 2014.

[139] *Metrics to Evaluate your Machine Learning Algorithm | by Aditya Mishra | Towards Data Science*. Accessed: Jul. 9, 2021. [Online]. Available: https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234

[140] M. Grandini, E. Bagli, and G. Visani, "Metrics for multi-class classification: An overview," Aug. 2020, *arXiv:2008.05756v1*.

[141] A. M. Carrington, D. G. Manuel, P. W. Fieguth, T. Ramsay, V. Osmani, B. Wernly, C. Bennett, S. Hawken, M. McInnes, O. Magwood, and Y. Sheikh, "Deep ROC analysis and AUC as balanced average accuracy to improve model selection, understanding and interpretation," Mar. 2021, *arXiv:2103.11357v1*.

DEEPALI ARUN BHANAGE received the master's degree in computer engineering from the Sinhgad Institute of Technology, University of Pune. She is currently pursuing the Ph.D. degree with the Symbiosis Institute of Technology, Symbiosis International (Deemed University), Pune. She is employed as an Assistant Professor with the PES's Modern College of Engineering, Pune. Her research interests include IT infrastructure monitoring, machine learning, deep learning, and natural language processing.

AMBIKA VISHAL PAWAR received the Ph.D. degree from Symbiosis International (Deemed University), Pune, India. She is currently an Associate Professor with the Computer Science and Information Technology Department, Symbiosis Institute of Technology, Symbiosis International (Deemed University). She has more than 19 years of experience as an Academician and more than ten years as a Researcher. She has published 47 research paper publications in international journals/conferences and one book published by Taylor & Francis, CRC Press. According to Google Scholar, her articles have 135 citations, with an H-index of six and an i10-index of four. Her research interests include security and privacy solutions using blockchain and AIMLDL technologies.

KETAN KOTECHA has worked as an Administrator with Parul University and Nirma University and has several achievements in these roles to his credit. He has expertise and experience in cutting-edge research and AI and deep learning projects for more than the last 25 years. He has pioneered education technology. He is a Team Member for the nationwide initiative on AI and deep learning skilling and research named Leadingindia.ai initiative sponsored by the Royal Academy of Engineering, U.K., under the Newton Bhabha Fund. He currently heads the Symbiosis Centre for Applied Artificial Intelligence (SCAAI). He is considered a Foremost Expert in AI and aligned technologies. He is also with his vast and varied experience in administrative roles. He has published widely in several excellent peer-reviewed journals on various topics ranging from education policies and teaching-learning practices and AI for all.