Spring 3-4-2021

Review and Analysis of Failure Detection and Prevention Techniques in IT Infrastructure Monitoring

Deepali Arun Bhanage
Symbiosis Institute of Technology, Symbiosis International University, Pune, Maharashtra, India,
deepali.bhanage.phd2019@sitpune.edu.in

Ambika Vishal Pawar
Symbiosis Institute of Technology, Symbiosis International University, Pune, Maharashtra, India,
ambikap@sitpune.edu.in

K Kotecha
Symbiosis Centre for Applied Artificial Intelligence, Symbiosis International, Pune, India,
director@sitpune.edu.in

Review and Analysis of Failure Detection and Prevention Techniques in IT Infrastructure Monitoring

Abstract:

Maintaining the health of IT infrastructure components for improved reliability and availability is a research and innovation topic for many years. Identification and handling of failures are crucial and challenging due to the complexity of IT infrastructure. System logs are the primary source of information to diagnose and fix failures.

In this work, we address three essential research dimensions about failures, such as the need for failure handling in IT infrastructure, understanding the contribution of system-generated log in failure detection and reactive & proactive approaches used to deal with failure situations.

This study performs a comprehensive analysis of existing literature by considering three prominent aspects as log preprocessing, anomaly & failure detection, and failure prevention.

With this coherent review, we (1) presume the need for IT infrastructure monitoring to avoid downtime, (2) examine the three types of approaches for anomaly and failure detection such as a rule-based, correlation method and classification, and (3) fabricate the recommendations for researchers on further research guidelines.

As far as the authors' knowledge, this is the first comprehensive literature review on IT infrastructure monitoring techniques. The review has been conducted with the help of meta-analysis and comparative study of machine learning and deep learning techniques. This work aims to outline significant research gaps in the area of IT infrastructure failure detection. This work will help future researchers understand the advantages and limitations of current methods and select an adequate approach to their problem.

Keywords: IT Infrastructure, log analysis, Failure Detection, Failure Prediction, ITIL

1. Introduction

IT infrastructure is the composition of IT components required by users and businesses for the activities and services to support business functions. It deals with hardware, software, services and network resources necessary for the operation and management of IT environments. There is a need for IT service management which can help in designing, delivering, creating, supporting and managing the life process of IT assets and services (“IT service management (ITSM): process, benefits, ITSM vs ITIL, best practices & metrics,” n.d.). This management depends on the understanding of components in IT infrastructure and associated tasks.

The failure in IT infrastructure assets has been the topic of research and innovation for many years. Failure rates are tremendous, even though many research has been done in this area. Several industry surveys show that there are significant losses due to IT infrastructure downtime. As per the international data corporation survey, the moderate cost for unplanned network downtime in companies is $5,600p per minute, which is as stated in Gartner ("The Cost of Downtime - Andrew Lerner," n.d.). Such monetary and non-monetary losses demonstrate that there is a need to handle IT infrastructure failures.

Various reasons are presented due to which IT infrastructure can fail and result in downtime. Performance bottlenecks, software or hardware failure, file system malfunction, connection loss, software issues such as application bugs or errors, insufficient allocated resources and cluster managing system error etc., are a few examples of causes of failures. Listed conditions can occur in any IT infrastructure components and propagate failure in the whole infrastructure.

Two conventional approaches to handle the failures in IT infrastructure are reactive and proactive (Tan & Gu, 2010). In the reactive approach, corrective action is taken after the failure happens. In this approach, even if quick action is taken to find the cause of the error and promptly handle the failures, it causes downtime. Thus, there will be system downtime in the reactive approach, which is generally undesirable for continuously running applications. Whereas, in a proactive approach, proactive actions are taken before failure occurs to avoid it; thus, it prevents the downtimes and associated losses. The proactive approach works on the prediction that can forecast system failures so that corrective action will be taken to avoid the failure. This approach offers better reliability by preventing downtime.

IT infrastructure monitoring has become a challenging task due to increased complexities in IT infrastructure and its utilization. Any failure for a small amount of time leads to significant losses to an organization. Thus it is foremost essential to avoid such failure conditions.

When any system in IT infrastructure does not work as it intended to function, it is called a system failure. Since the early days of computers, system-generated logs used to handle such failures in the systems (Pecchia, Weber, Cinque, & Ma, 2020). The majority of the research work has considered system logs as the primary source of data for any system as it records the states and individual runtime behavior (Fu, Ren, Mckee, Zhan, & Sun, 2014). Traditionally administrators were detecting system anomalies and root causes of failures by understanding the status and behavior of the system using generated log information. The authors (R. Ren et al., 2019) reveals that system log analysis is an effective and comprehensive method for self-regulating IT infrastructure management, monitoring, intervention, failure prediction and root cause diagnosis. The authors (Pecchia et al., 2020) (Zou, Qin, & Jin, 2016) suggested identifying keywords that denote failures such as error, fault, unavailable,

fatal etc., from unstructured log data is a common approach used for the detection of failure. The authors (Jain, Singh, Chandra, Zhang, & Bronevetsky, 2009) uncover that the detection of system anomalies getting provocative by virtue of an increase in scale and complexity. Thus it shows the essentiality of an automated system that can detect failures and perform self-correction actions in IT infrastructure.

As IT Infrastructure logs provide the information about each component's status and record the system operational changes such as starting or stopping services, software configuration modifications, software execution errors and hardware faults, and so on. The administrator can use this information to understand system behavior and detect anomalies. Various systems generate log information in different formats and record other pieces of information.

Researchers have explored various types of log for analysis purpose which includes activity log (Saadatfar, Fadishei, & Deldari, 2012), console log (K. Zhang et al., 2016) (Das, Mueller, Hargrove, Roman, & Baden, 2019), event log (Pitakrat, Grunert, Kabierschke, Keller, & Van Hoorn, 2014), exception log (Y. Yuan, Shi, Liang, & Qin, 2019), fault log (Zou et al., 2016), job log, ALPS log, big data log (Wu et al., 2019), RAS log (Zheng, Lan, Gupta, Coghlan, & Beckman, 2010), message log (Chuah et al., 2019), network log (Bertero, Roy, Sauvanaud, & Tredan, 2017) system log (Kimura, Watanabe, Toyono, & Ishibashi, 2019) (Fu et al., 2012) (Meng, Liu, Zhu, et al., 2019) (R. Ren et al., 2019) (Gainaru, Cappello, Fullop, Trausan-Matu, & Kramer, 2011) (M. Wang, Xu, & Guo, 2018) (M. Du, Li, Zheng, & Srikumar, 2017) (Lu, Wei, Li, & Wang, 2018) (X. Zhang et al., 2019) and transactional & operational log (Jia, Yang, et al., 2017) etc.

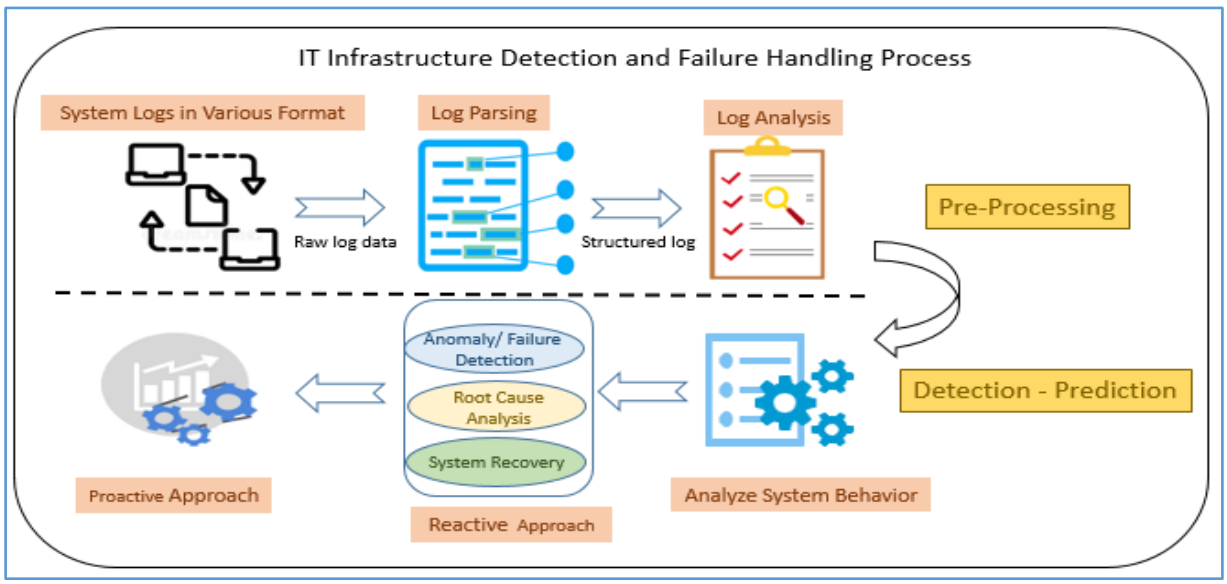

Figure 1 Infrastructure Failure Detection and Prevention Pipeline

Figure 1 shows the pipeline of IT infrastructure failure detection and prevention process. The first step in this is to collect logs from the various systems. Collected logs are available in various formats, thus requiring processing and converting from unstructured to structured form by reducing noise and duplicate data. It also performs data abstraction considering the similarity and relevance of information in log data. The next step is for log analysis to make the log more readable and understandable. With log analysis, one can detect anomaly or failure in the IT infrastructure components. Two types of actions can be taken to handle identified failures, reactive to revert the effect of failure and proactive to prevent failure condition in future by predicting it.

1.2 Contribution of this work

Many IT companies are working in the field of IT infrastructure monitoring to manage and optimize IT infrastructure and ensure continuity in service. BMC TrueSite, IBM Tivoli, BladeLogic are some of the popular IT infrastructure monitoring software currently available in the market. Also, these companies are using different mechanisms to monitor IT infrastructures. Among other mechanisms used, log analysis is one of the popular mechanisms adopted by many companies. In the recent past few years, several new approaches, as well as tools and techniques, are being suggested by researchers to deal with the IT infrastructure failure problems. Many researchers have endeavored to identify failures in IT infrastructure components such as network, supercomputers, cloud system, distributed system, hardware, applications, etc.

This literature review focuses on existing research done in IT infrastructure monitoring and takes it ahead to improvise existing work. In this systematic study, we explore the failure handling and prevention techniques to maintain the health of IT infrastructure. This comprehensive study also concentrated on an analysis of approaches explored by several researchers. We also demonstrate the exhaustive meta-analysis of various components like IT infrastructures, datasets, methodologies etc., utilized in the existing literature. The literature study also scrutinized the tools and techniques based on derived results to pinpoint the worthwhile research gaps.

The rest of the paper is organized as follows. Section 2 brief about related work done in the IT infrastructure monitoring area. Section 3 illustrates the nature of the log data, which is used in the IT infrastructure monitoring study. Section 4 carried out a detailed discussion on scholarly publication in the existing literature. Section 5 illustrates the meta-analysis of studied scholarly publications. Section 6 gives an overview of automated tools studied during the literature review. Section 7 represents the comprehensive analysis of existing literature. Section 8 exchange views on concluding remarks with future work.

2. Methodology Framework for Literature Review

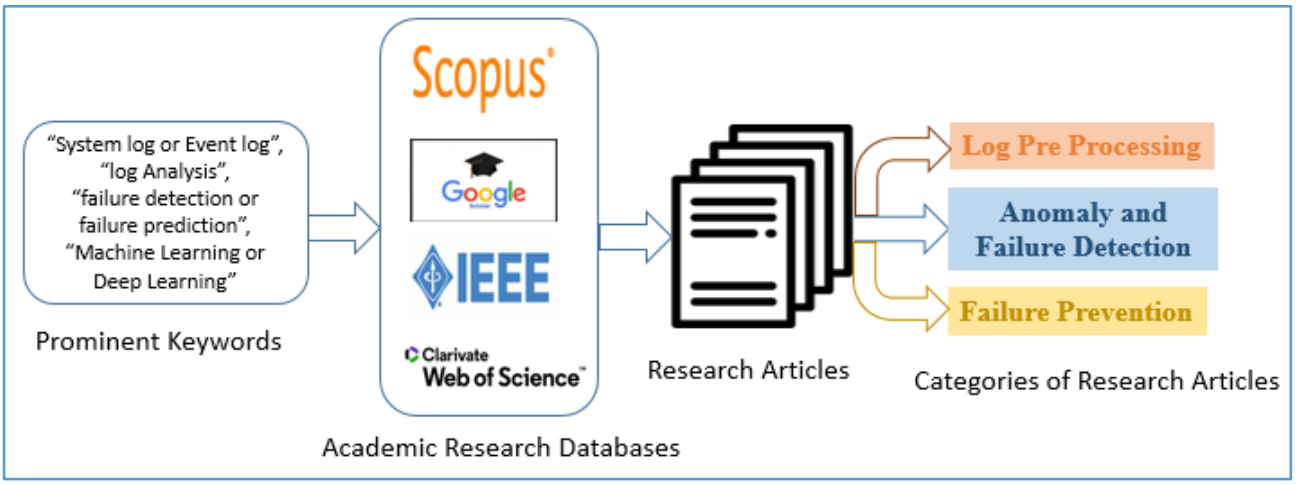

Figure 2 Methodology Framework for Literature Review

Definite scholarly articles are collected from various databases like Scopus, Google Scholar, IEEE, Science Direct etc. We designed a search query using appropriate keywords such as "system log or event log", "log Analysis", "failure detection or failure prediction", "machine learning or deep learning", etc. In this literature review, we studied around 100 research publications. The scholarly articles and the authorized web links are also referred to gather information about information technology service management (ITSM). All research articles were studied carefully and further classified into three categories based on the work's purpose. Log preprocessing, anomaly & failure detection and failure prevention categories are discussed in detail in upcoming sections.

3. Nature of Log Data

All the components in IT infrastructure generate logs that contain messages from several modules. A software developer writes predefined logging statements in the source code of the software to generate logs at the time of execution of the system. According to (X. Zhang et al., 2019), every 58th line in the source code is for the log. Thus every system has a log file in its format. Logs get recorded in the system when a noteworthy event occurs. As logs get recorded at system runtime, they are the primary source of information. Although logs look like plain text, it has some standard components such as timestamp, level of the log, unique ID, variables and exceptions inside the log. Where timestamp, log level and ID are considered log header and rest of the points as part of log messages. In log messages, few entries are static fields, whereas few are dynamic. In log message, static fields are written by a developer in source code and dynamic field updates at runtime. Log level plays a critical role in the case of troubleshooting.

Table 1 Common Logging Levels

Level	Description
FATAL	About to abort
ERROR	Failure
WARN	Unusual situation
INFO	Normal Behavior and Milestones
DEBUG	Diagnostic information
TRACE	Fine-grained information
ALL	Record everything
OFF	Don't log anything

Table 1 shows different levels of logs with a description. Considering separate application and the purpose to record logs, they are generated in various formats.

Concerning the various logs, observation says, logs are the combination of characters, numbers and special symbols. These are not in a readable form, or one cannot retrieve proper meaning out of it. Thus, we need first to convert such unstructured log to structured log.

Figure 3 gives the decomposition of elements in the sample windows log. All the logs are not having precisely the same format, but few features are standard. It is possible to derive information about the date, time, log level, component, and contents from any log message. With the help of understanding these elements of logs, one can find the event template.

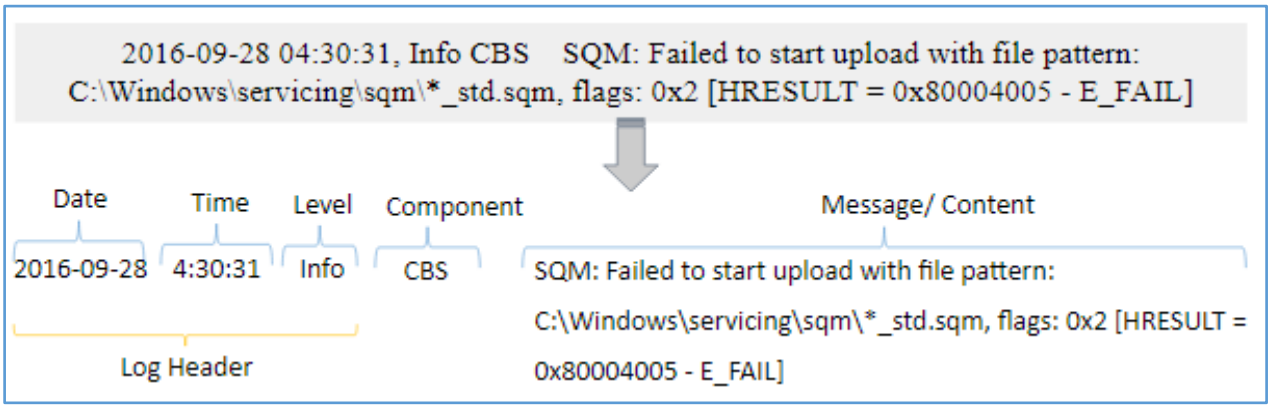

Figure 3 Elements of Example Windows Log

4. Study of Scholarly Publications

This literature review focuses on three significant phases of the failure handling process: log preprocessing, anomaly & failure detection, and failure prevention. The method of log preprocessing involves two steps. The log parsing based on probability of occurrence (Basak & Nagesh, 2016), NPL features (Aussel, Petetin, & Chabridon, 2018) and filtering are the commonly used techniques by researchers. Clusters of relevant logs formed in log analysis can help for a better understanding of log data. In an anomaly or failure detection phase, researchers focused on machine learning

(Bronevetsky, Laguna, De Supinski, & Bagchi, 2012) (Otomo, Kobayashi, Fukuda, & Esaki, 2019), LSTM (M. Du et al., 2017)(M. Wang et al., 2018)(X. Zhang et al., 2019) and correlation techniques (Chuah et al., 2019) (Y. Yuan, Shi, et al., 2019) (Farshchi, Schneider, Weber, & Grundy, 2018). Concerning the existing literature, failure prevention is possible by predicting fault propagating conditions like event prediction (Fu et al., 2012) (Gainaru et al., 2011), hardware maintenance prediction (J. Wang, Li, Han, Sarkar, & Zhou, 2017), calculate remaining useful time (Shen, Wan, Lim, & Yu, 2018) (Chaves, De Paula, Leite, Gomes, & MacHado, 2018) and root cause analysis (Konno & Défago, 2019).

4.1 Log Pre Processing

The generated log is enormous data that is ambiguous, unstructured, incomplete and duplicate. To make better usage of this data, first, we need to process it. This preprocessing includes two steps. The first step is converting a raw and unstructured log to a structured log by removing noise and duplicate entries, which is called a parsing process. After data abstraction, the next step is log analysis to form clusters of similar types of messages. This classification is helpful for anomaly or failure detection and further for doing the root cause analysis. The process of classification is also called log mining.

Figure 4 gives an overview of log preprocessing techniques. After critical analysis of research articles, observation is that leading categories of methods in log preprocessing approaches are clustering, NLP and filtering. Most of the researchers have used different features of log for clustering, such as the probability of appearance of words, frequency of occurrence and pair of messages which occur together etc. The authors suggested applying natural language processing techniques when you consider log messages as plain text. With the help of NLP techniques, log messages are converted into a meaningful sentence or represented in the form of a vector for further analysis. Filtering techniques used for log abstraction by removing unwanted entries.

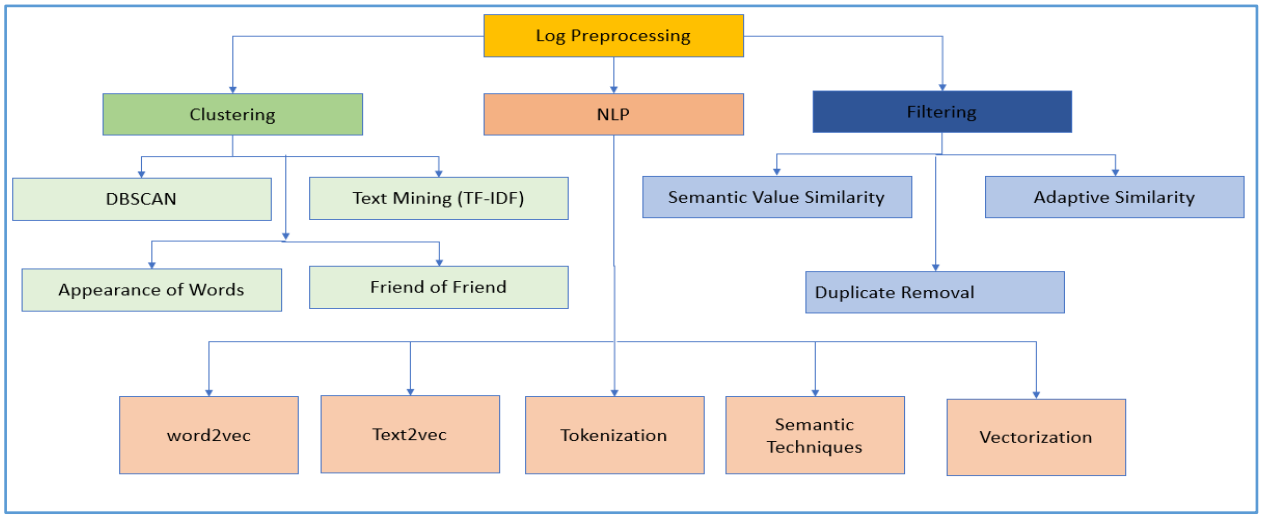

Figure 4 Log Preprocessing Techniques

The existing log analysis methods demand improvement in data about resources used, faults related to timestamp, useful and necessary to identify problems in generated logs (W. Yuan, Lu, Sun, & Liu, 2020). Some researchers have identified the errors in the generated log, such as inappropriate log messages, missing logging statements, inadequate log level, log library configuration issues, runtime issues, overwhelming logs, and log library changes (Hassani, Shang, Shihab, & Tsantalis, 2018). Therefore before selecting the log data for preprocessing, it is necessary to check its quality.

The authors (Jain et al., 2009) stated that to derive necessary information and make it readable from a huge supercomputer log is possible by decoding. Still, it may result in the loss of valuable information. Conjunctive, disjunctive, and markovian data filtering approaches were used by the authors (Basak & Nagesh, 2016) (Huang, Ke, Wong, & Mankovskii, 2010) reduce 30% to 50% hard disk log data by considering the usefulness of log message. According to (Oliner & Stearley, 2007), the abstraction will help detect the root cause of failure, establish a correlation among logs and classify various failures. In (El-Masri, Petrillo, Guéhéneuc, Hamou-Lhadj, & Bouziane, 2020), the authors investigate the performance of different abstraction techniques based on seven quality aspects such as mode, coverage, delimiter independence, efficiency, scalability, system knowledge independence, and parameter tuning effort. The authors (Aussel et al., 2018) (Amato, Cozzolino, Mazzeo, & Moscato, 2019) concluded that log parsing is also possible through simple NLP techniques, which are efficient over the rule-based approach. Also, one can focus on advanced NLP techniques to process complex log to get a relevant result. Authors (Tak, Park, & Kudva, 2019), (Kobayashi, Otomo, Fukuda, & Esaki, 2018) stated that converting the log data in the time series data is one of the essential techniques used for log preprocessing. The researchers (Z. Li, Davidson, Fu, Blanchard, & Lang, 2018) (Z. Li, Davidson, Fu, Blanchard, & Lang, 2019) made use of a System Log Event Block Detection (SLEBD) framework to identify event blocks which can help in behavior analysis based on events. In the study of (Pettinato, Gil, Galeas, & Russo, 2019), the Latent Dirichlet Allocation algorithm used to discover latent topics in messages of ALMA telescope system log events. A dynamic matrix factorization approach (dynamic MF) has been proposed by (Sorkunlu, Anh Luong, & Chandola, 2019) to reduce the dimension of resource usage data and visualize it at the node-specific level. Reduction in sizes and simple visualization will help in anomaly detection. In the paper (Dua, Choudhury, Rajanikanth, & Choudhury, 2019), authors use the C programming language to assign tagged values of virtual machine log data. This tagged Syslog is helpful for log classification or correlation, which will help in the identification of failure.

4.2 Anomaly and Failure Detection

The abnormality in the system leads to the fault, which results in failure. An anomaly is an unexpected behavior of the system, which may lead to failure. Failure is the condition opposite to success.

By exploring the existing literature, observation is made that it will be available for anomaly or failure detection after analysis of log data. Most of the researchers focused on machine learning algorithms, deep learning algorithms and correlation methods. Much research has been done in this area, but existing systems necessitate finding a correlation between the alerts and events to reduce the false alarms (Le & Zincir-Heywood, 2020).

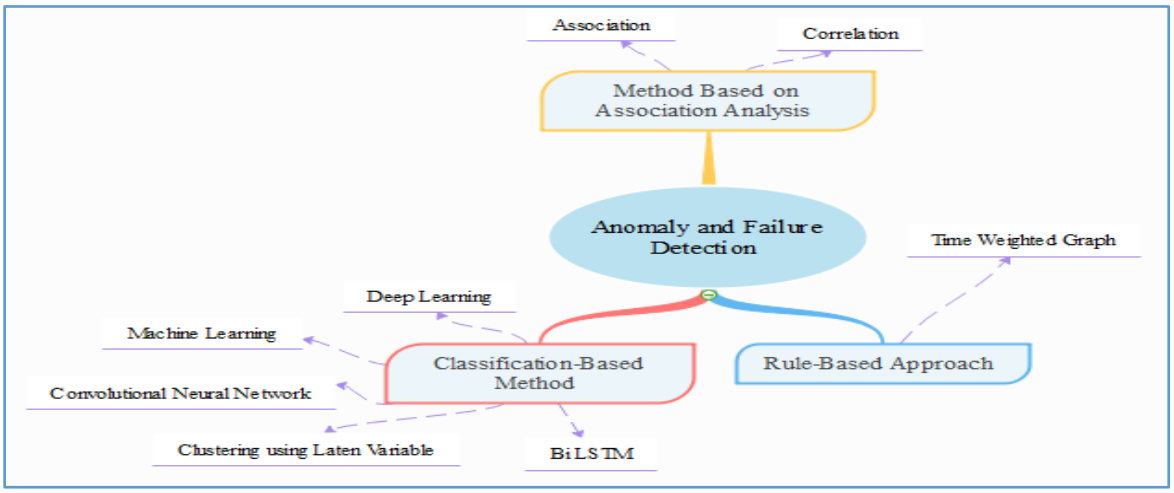

Figure 5 Classification of Anomaly and Failure Detection Techniques

Figure 5 illustrates the classification of anomaly and failure detection approaches used in current research work. After a rigorous analysis of research articles, we can say predominantly detection approaches classified into three categories, such as rule-based approach, a method based on association analysis and classification based methods.

4.2.1 The rule-based approach:

This approach follows the guiding principles to express knowledge and the rules stated by experts in advance. In LogSed (Jia, Yang, et al., 2017) Black-box method is used to recognize anomalous runtime behaviors from the transactional log and operational log of the cloud. The authors (Nandi, Mandal, Atreja, Dasgupta, & Bhattacharya, 2016) (Jia, Chen, et al., 2017) have claimed 80% precision and recall rate by using time-weighted control flow graphs (TCFG).

4.2.2 Correlation and Association based approach:

This approach determines the correlation between various system features that derive association rules and adopt them for anomaly or failure detection. The author compares correlation data with historical data of the open stack system for failure detection. In the work of (Farshchi et al., 2018), a regression-based approach proposed to encounter anomalies in the execution of amazon DevOps operations for rolling upgrade operations. (B, Cruzes, Angulo, & Fischer-h, 2016) They designed a LADT (lightweight anomaly detection tool) that raises an anomaly alarm when the correlation coefficient value between the node-level and VM-level metrics drops below a threshold level. This is

useful to represent the relation between cloud operation behavior and the changing states of cloud resources.

In the research of (Lin, Zhang, Lou, Zhang, & Chen, 2016), the comparison between the newly generated log cluster and knowledge base performed to detect the problem in online service systems if the cluster is not available in the knowledge base to take help from an administrator to examine manually. According to (Di, Guo, Pershey, Snir, & Cappello, 2019), the authors calculated the meantime to interruption (MTTI) 3.5 days for the whole Mira system by performing RAS mapping events and job failure data.

4.2.3 Classification based approach:

In the existing literature, classes are labelled based on the various features of log data like time stamp, length of the message, level of the message, type of error etc. and outlier detection treated as an abnormality.

Even though some researchers explore correlation analysis for anomaly and failure detection, according to (Zou et al., 2016), the classification approach shows improvement in results. The authors also uncovered that identifying the root cause is possible by understanding the current status of the cloud with the help of the classification of the fault log. As per the conclusion of (Bertero et al., 2017), one of the critical factors for anomaly detection is the HPC system's stress behaviour. As per the research of (Meng, Liu, Zhang, et al., 2019) (Jin Wang et al., 2020) use of NLP techniques for preprocessing followed by classifiers reduces the computational time and gives an excellent F1 score for anomaly detection. (Meng, Liu, Zhu, et al., 2019) (M. Wang et al., 2018) Explores a deep learning-based approach using LSTM for anomaly detection using exception log datasets for HDFS system. In addition to that (X. Wang, Wang, Zhang, Jin, & Song, 2019) stated, upgraded LSTM based abnormal behavior detection system is required to ensure the network system's regular operation, which can provide multi-dimensional warning information. In the research work of (M. Du et al., 2017) (X. Zhang et al., 2019), they proved that a deep learning approach gives better results than machine learning or correlation-based algorithms. Also (Lu et al., 2018) have demonstrated work on logkey2vec algorithm (CNN) based approach to earn superior and agile detection accuracy than MLP and LSTM on HDFS system logs. In this approach, log parsing performed directly without any application-specific information.

In the study (Chen, Singh, & Yajnik, 2012), (S. Du & Cao, 2015), the hierarchical clustering algorithm used to form clusters to identify anomalies based on its score, neglecting the incompleteness of logs. The authors stated that (Ahmad, Lavin, Purdy, & Agha, 2017) hierarchical temporal memory (HTM) gives excellent results on server metrics and online advertisements but not adequate for expressing temporal anomalies. One way for anomaly detection is by representing the

log data in time series format and processing it. Whereas in the DeepAnt tool (Munir, Siddiqui, Dengel, & Ahmed, 2019), the CNN approach identifies an anomaly in time series data.

By examining the challenges in getting or generating the labelled log data, (Borghesi, Bartolini, Lombardi, Milano, & Benini, 2019) (Ghiasvand, 2019) applied a Semi-supervised autoencoder based approach to learning the behavior of HPC systems.

4.3 Failure Prevention

If the user or administrator gets fault information and details about the failure before it happens, he will take corrective actions and avoid failure conditions. Fault detection is possible by merely finding the deviations in the regular system behavior. In order to handle the faults, the crucial thing is to identify the root cause and get details about location, time and fault information. Once the fault is detected, heal it by taking corrective actions.

Table 2 Failure Prevention state of the art summary

Ref & year	Area/ System Used	Data used	Methodology	Relevant Insights
(Zheng et al., 2010)	IBM Blue Gene/P system	Job log, RAS log	Genetic Algorithm	Precision and recall decreases with a growing lead time
(Gainaru et al., 2011)	HPC	System log	Correlation	Correlation chain between event to identify behavior
(Saadatfar et al., 2012)	Grid System	Activity log	Bayesian network (DM)	Job Failure prediction accuracy varies with selected features and training window size
(Fu et al., 2012)	Hadoop, HPC, BlueGebe/L	System log	Event Correlation Graph-based algorithm	Event prediction precision rates for event prediction is maximum 83.66%, 81.19% and 79.82%, respectively
(Gainaru, Cappello, Snir, & Kramer, 2012) (Gainaru, Cappello, Snir, & Kramer, 2013)	HPC	Event log	Signal Analysis Data Mining	The hybrid module gives better results still required to improve the recall rate. The system can discover about 50% of all failures.
(Pitakrat et al., 2014)	Blue Gene/L	Event log	Machine Learning	Human intervention required in the labelling event log.
(K. Zhang et al., 2016)	webserver, mailer server cluster	console log	Deep Learning (LSTM)	Deep learning outperformed machine learning in terms of PR-AUC, predictable interval and predictable frequency
(Yoo, Sim, & Wu, 2016)	Genepool scientific cluster	Job log	ML- Binary Classifier	Job-status prediction can help to reduce time, resource waste, and cost against failures

(J. Wang et al., 2017)	ATM	Error log	ML	Improved AUC by 3% to 5% due to the use of feature selection techniques.
(Farshchi et al., 2018)	Amazon Web Services	Operational Event log, Resources Matrix	Correlation & Regression	Injected fault detection possible with high precision and recall by stating the relation between cloud operation behavior and changing states of cloud resources
(Shen et al., 2018) (Chaves et al., 2018)	Hard Disk	SMART attributes	Random Forest	Deep learning can give more accurate results than a random forest or bayesian network
(Pitakrat, Okanović, van Hoorn, & Grunske, 2018)	Netflix's server		Bayesian network	Predict failures propagation path caused due to memory leak, system overload, and sudden node crash.
(Rawat, Sushil, Agarwal, & Sikander, 2018)	Virtual Machine	Time series data	ARIMA	The proposed approach can be used for the proactive fault tolerance technique
(R. Ren et al., 2019)	Cluster System	System log	Deep CNN	Event category prediction with 98.14% precision for classification
(Meng, Liu, Zhu, et al., 2019)	HDFS, BLG	System log	Deep Learning	Avoid false alarms using semantic information of log
(Gao et al., 2019)	Hard Disk	SMART attributes	ML	7% increase in recall rate
(Kimura et al., 2019)	Network	System log	ML	Pattern-based approach more efficient
(Das et al., 2019)	Cray systems	Job log, ALPS log, Console log	Time-Based Frame	2 min prediction lead time
(Wu et al., 2019)	Bigdata	Bigdata log	RNN-seq2seq algorithm	Never appeared logs in history cannot be predicted.
(Pal & Kumar, 2019)	Network	Network log	Ensemble learning	Ensemble learning gives better results than individual classification algorithms.
(Xiang, Huang, & Li, 2019)	Vending Machine	System log	ML- Binary Classifier	80% accuracy in terms of precision, recall, and F-measure using a two-stage predictive model.
(Y. Li et al., 2020)	Cloud	Times series data	ML – Random Forest	The system has not focused on the type of node failures and the root cause of failure.

In the existing research, till now, predictions are performed on hardware component failure, event failure, job failure etc. Also, systems implemented to predict maintenance time, remaining useful life of hard disk and stress in the network to maintain the health of the system. To improve the reliability of the system, traditionally check pointing and monitoring of the system techniques are in use.

Table 2 illustrates the summary of research articles studied under this literature review for failure prevention. Researchers have considered various systems in the literature to improve the reliability and availability of IT infrastructure components. Table 2 elaborates on research components used by researchers such as systems, the type of log data, the methodology used to deliver results and relevant insights, which explain the key points from several research articles that can contribute significantly to further research.

In the research work, (Zheng et al., 2010) applied a genetic algorithm on the RAS log to detect the location of failure in the IBM Blue Gene/P system with 0 to 600 seconds lead time. The study of (Saadatfar et al., 2012) identified the failure pattern, which promotes job failure prediction in the product grid. In this work, the authors used activity log mining to find the relation between workload characteristics and job failure. (K. Zhang et al., 2016). Focused their work on a deep learning approach to generate early failure warning signals in the web server and mailer server cluster. In the paper (Gainaru et al., 2013), the authors used data mining techniques to extract the pattern in log data and show a correlation between defined behavior. This hybrid approach gives better results than an individual policy. The authors (Gao et al., 2019) conclude that the nearest neighbor algorithm based on the density matrix offers 7% more accuracy than unsupervised algorithms in disk failure prediction. Authors (Kimura et al., 2019) explores their work on patterns of log messages and trouble ticket data to predict network failures using supervised machine learning algorithms. The authors (Pal & Kumar, 2019) conclude that ensemble learning outperforms than the individual classification algorithm. Among the several tools, (Choudhary & Singh, 2013) authors tried the hidden Markov model approach to analyze and predict failures in a Hadoop cluster with 91% accuracy for two days in advance. According to (Pitakrat et al., 2018), a failure propagation path will help more avoid failure conditions in rapidly changing systems and failure prediction. (Rawat et al., 2018) Proposed time series techniques to predict future failure points in a virtual machine.

Furthermore, this approach can be useful for dynamic fault tolerance by detecting the type of node failure and the root cause of it (Y. Li et al., 2020). Concerning research done in (Ozcelik & Yilmaz, 2016), an appropriate combination of hardware and software can improve the quality of software. The presence of multiple hardware in the system defends the online failure prediction instead of single.

As per investigated literature, event prediction in the HPC system is possible with observing the behavior. In the work of (Gainaru et al., 2011), log at different time windows is considered, and (Pitakrat et al., 2014) proposed a machine-learning algorithm to identify the pattern of events that often appear together. Also, (Wu et al., 2019) targeted the Seq2seq algorithm to predict an event that causes IoT node failure in a selected time window.

As per the literature study, predicting the correct time of maintenance is one way to prevent failure in the hardware devices. Also, prediction of exact maintenance time of ATM (J. Wang et al., 2017) and vending machine (Xiang et al., 2019) demonstrated by classification of the event log and failure log, respectively.

Some research studies (Shen et al., 2018) work to calculate the remaining useful time for the hard disk with the help of self-monitoring, analysis and reporting technology (SMART) attribute classification using Bayesian network and random forest algorithms. Their finding also suggests that SMART parameters can help check the health of the hard disk (Chaves et al., 2018).

Future job prediction is possible using data mining (Saadatfar et al., 2012) and machine learning (Yoo et al., 2016) to reduce the downtime in the grid system.

Concerning ("Root Cause Analysis (RCA) for IT – BMC Blogs," n.d.) we can say root cause analysis is not only helpful to pinpoint factors that contribute to the problem but also to resolve the issue as fast as possible. Failure conditions can be prevented by avoiding known causes of it. For example, the service management quality of cloud computing can be improved by finding the root cause of failure using event logs in the in-memory time-series database stated by (Konno & Défago, 2019). In the study (Lu et al., 2017), the spatial-temporal analysis was conducted on the execution log and garbage log for root cause analysis in the spark system. Wordcount, Kmeans and PageRank algorithms were applied on spark log for CPU, memory, network and disk features.

Table 3 Items and Techniques used for Root Cause Analysis in Existing Literature

Author	Items Required for Root-cause Diagnosis	Techniques used
Lu et al., 2017	Execution Log and Garbage Collection Log	Weighted Factor
Weng et al., 2018	Metrics Data of Services and Resource Utilization	Similarity Score
Yuan et al., 2019	Log Event Sequence and Cloud Service Behavior	Vectored Event Sequence
Konno & Défago, 2019	Metrics and Event Logs	Event-Driven Active Monitoring

Another approach proposed by (Weng, Wang, Yang, & Yang, 2018) for root cause analysis is to calculate similarity score based on metrics data of services and resource utilization, giving 15%- 71%

improved precision. The researchers (Y. Yuan, Anu, Shi, Liang, & Qin, 2019) explored the method which can learn from experience and automatically decode cloud service behavior based on user operations to determine the cause of the anomaly.

Recovering from the failure condition is the reactive approach for failure handling. Check pointing is the traditional technique used for failure recovery. In the study (Tiwari, Gupta, & Vazhkudai, 2014), the authors proved that the lazy check pointing technique could significantly reduce the I/O overhead and compute resource wastage helps to prevent the occurrence of failure conditions. The authors (Qi, Tsai, Li, Zhu, & Luo, 2017) advised that parallel analysis of workflow in the amazon cloud is advantageous and assist in workflow recoveries. (Jha et al., 2018) Pinpointed the issue of failure occurs during the recovery process. Based on this study, the system is designed to identify interconnected failures and recovery procedures, which will help to understand the category of failure and propagation during recovery.

5. A meta-Analysis of Studied Scholarly Articles

This section presents the meta-analysis of studied scholarly publication from literature work. Meta-analysis is carried out based on four components which are derived from the rigorous analysis of respective articles. The list of components, sub-components, properties and related descriptions are specified in table 4.

Table 4 List of Components and their properties used for Meta-Analysis of Scholarly Publications

Component	Label	Property	Description
Infrastructure	I1	Supercomputer	Potent computers with great speed and memory
	I2	Distributed System	Numerous components spatially separate but connected in the network
	I3	Cloud System	On-demand computer system resources over the internet
	I4	Network	Infrastructure components connected to share resources
	I5	Hardware	The physical component of the computer system
	I6	Other	Any other system rather than listed above
Dataset	D1	log	Complete information about all executed operations
	D2	Time Series Data	Time attached to each value of the information sequence
	D3	Other Metrics	Temporal and spatial data about the system
Category of work	C1	Preprocessing	Data cleaning and analysis to reduce the size
	C2	Detection	Detection of anomaly, failure or error to deal with
	C3	Prediction	Prediction to avoid failure conditions
	C4	Recovery	Recovery to cover damage due to downtime
Methodology used	M1	Clustering	Grouping set of logs based on similarity
	M2	NLP	Semantic analysis of log considering it as standard text
	M3	Filtering	Remove unimportant log to reduce the size
	M4	Rule-Based	The predefined set of rules forms the knowledge
	M5	Correlation	Find the relation between logs and various records
	M6	Data Mining	Derive useful data to detect abnormal execution
	M7	Machine Learning	Train system to detect or predict abnormal conditions automatically
	M8	Deep Learning	Analyze the massive amount of data for prediction

Table 5 Meta-Analysis of Scholarly Publications

Author	Year	Infrastructure						Dataset			Category				Methodology							
		I1	I2	I3	I4	I5	I6	D1	D2	D3	C1	C2	C3	C4	M1	M2	M3	M4	M5	M6	M7	M8
Zheng et al.,	2010	✓						✓		✓			✓					✓				
Adhianto et al.,	2010		✓					✓				✓	✓	✓							✓	
Chuah et al.,	2011	✓						✓		✓		✓							✓			
Saadatfar, Fadishei and Deldari,	2012		✓					✓					✓							✓		
Gainaru et al.,	2012	✓						✓					✓							✓		
Fu et al.,	2012	✓	✓					✓					✓						✓			
Fu et al.,	2014	✓	✓					✓					✓							✓		
Du and Cao,	2015		✓					✓		✓		✓			✓							
Zhang et al.,	2016			✓				✓					✓									✓
Zou, Qin and Jin,	2016			✓				✓				✓									✓	
Gurumdimma et al.,	2016	✓						✓		✓		✓							✓			
Yoo, Sim and Wu,	2016	✓						✓					✓								✓	
Nandi et al.,	2016		✓					✓				✓						✓				
Lin et al.,	2016		✓					✓				✓			✓							
Ozcelik and Yilmaz,	2016					✓				✓			✓								✓	
Wang et al.,	2017					✓		✓					✓								✓	
Jia et al.,	2017			✓				✓				✓						✓				
Du et al.,	2017		✓	✓				✓				✓										✓
Jia et al.,	2017			✓				✓				✓						✓				
Aussel, Petetin and Chabridon,	2018		✓					✓			✓		✓			✓					✓	
Farshchi et al.,	2018			✓				✓		✓		✓							✓			
Chaves et al.,	2018					✓				✓			✓								✓	
Shen et al.,	2018					✓				✓			✓								✓	
Rawat et al.,	2018			✓					✓				✓									
Liu et al.,	2018			✓				✓				✓									✓	
He et al.,	2018			✓				✓		✓		✓							✓			
Di et al.,	2018	✓						✓			✓						✓		✓			
Otomo et al.,	2019				✓			✓				✓										✓
Zhang et al.,	2019		✓					✓			✓	✓										✓
Chuah et al.,	2019	✓						✓		✓		✓							✓			
Ren et al.,	2019	✓	✓					✓					✓									✓
Meng et al.,	2019	✓	✓					✓			✓	✓				✓						✓

Pettinato et al.,	2019					✓		✓			✓									✓
Kimura et al.,	2019				✓			✓					✓							✓	
Charapko et al.,	2019		✓											✓							
Wu et al.,	2019						✓	✓					✓								✓
Das et al.,	2019	✓						✓	✓				✓							✓	
Xiang, Huang and Li,	2019					✓		✓					✓							✓	
Munir et al.,	2019			✓					✓			✓									✓
Ghiasvand,	2019	✓						✓				✓								✓	
Wang et al.,	2019				✓			✓				✓									✓
Yuan et al.,	2019			✓				✓					✓						✓		
Roumani and Nwankpa,	2019			✓					✓				✓							✓	
Tak, Park and Kudva,	2019			✓				✓			✓					✓			✓		
Borghesi et al.,	2019	✓						✓				✓									✓
Meng et al.,	2019				✓			✓			✓	✓				✓				✓	
Pecchia et al.,	2020						✓	✓			✓										
Wang et al.,	2020	✓						✓			✓	✓				✓				✓	
Zhang et al.,	2020	✓	✓			✓		✓			✓						✓				
Li et al.,	2020			✓					✓			✓								✓	

5.1 Machine Learning Techniques Used

Figure 6 represents the machine learning techniques used in the studied literature. In the research work of (Aussel et al., 2018) (Y. Yuan, Shi, et al., 2019) (Jin Wang et al., 2020), authors compared results of the various classifiers for anomaly and failure detection. At the same time, authors (J. Wang et al., 2017) concluded that the hybrid approach is more efficient than any individual forecasting model. In the research work of (Liu, Lv, Ma, & Yao, 2018), the authors concluded that the semi-supervised one-class support vector machine (OCSVM) method derives better performance on the unbalanced training dataset.

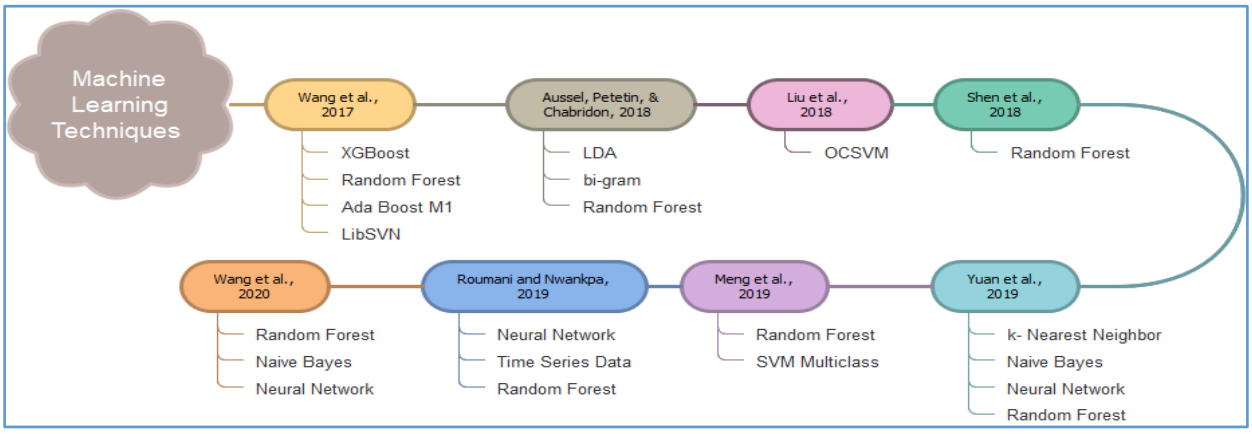

Figure 6 Machine Learning Techniques used in Existing Literature

The authors (Roumani & Nwankpa, 2019) suggested that machine learning and time-series methods are helpful to predict incidents in cloud systems. The recommended approach was tested on Netflix and Hulu without considering unreported incidents. Authors (Shen et al., 2018) performed self-monitoring, analysis and reporting technology (SMART) attribute classification using Bayesian network and random forest algorithms to calculate the remaining useful time for the hard disk.

5.2 Deep Learning Techniques Used

Figure 7 represents the deep learning techniques used in the studied literature. Considering the rapid increase in the volume of log data, deep learning techniques are more useful for training the detection or prediction model. Many researchers have claimed the efficiency of deep learning techniques in case of failure detection or prediction. The authors (R. Ren et al., 2019) findings suggested that deep learning approaches can provide great insights for understanding Hadoop and Bluegene/L logs by suppressing sensitive information about the business in event category prediction. In the research of (Otomo et al., 2019), authors performed mapping of time series data with latent variables, forming clusters to identify deviations. To get better results in the research work (Y. Ren et al., 2020), (Xie et al., 2020), a combination of machine learning and the statistical learning method considered for conformity measurement. In conformal prediction, classification is based on p-value; this is not in the format of 0 or 1. In the paper (Bronevetsky et al., 2012), the research was conducted to study the limitation of machine learning models in fault detection. Authors proved that a combination of classification and information on the abnormality gives an improvement in location and time period detection accuracy. By examining the challenges in generating the labelled log data, (Borghesi et al., 2019) (Ghiasvand 2019) applied a semi-supervised autoencoder-based approach to learning the behavior of HPC systems.

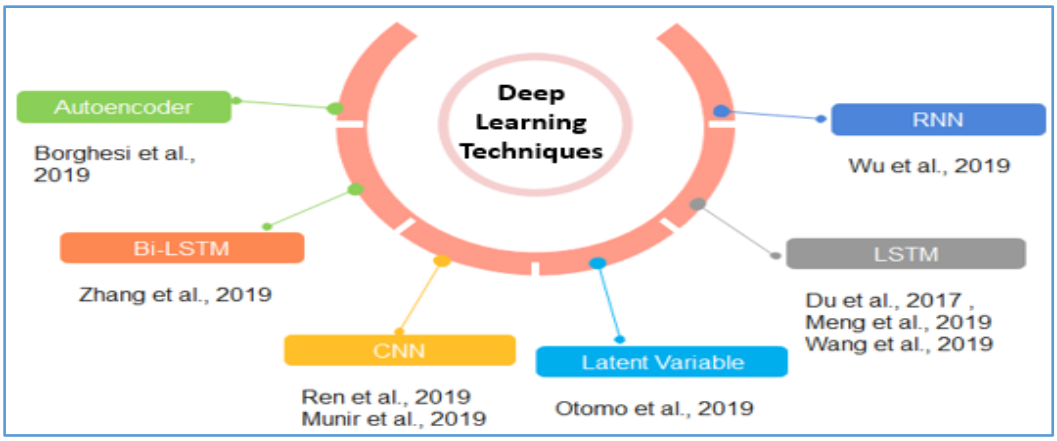

Figure 7 Deep Learning Techniques used in Existing Literature

6. Automated Tools Studied from Existing Literature

Table 6 Summary of tools in Studied Literature

Tool	Log Parsing	Log Analysis	Detection	Prediction	Recovery
POP (P. He, Zhu, He, Li, & Lyu, 2018)	✓				
UiLog (Zou et al., 2016)	✓	✓			
LogSed (Jia, Yang, et al., 2017)			✓		
DeepLog (M. Du et al., 2017)		✓	✓		
CORRMEXT (Chuah et al., 2019)			✓		
Loganomaly (Meng, Liu, Zhu, et al., 2019)	✓			✓	
Logmaster (Fu et al., 2012)		✓		✓	
Doomsday (Das et al., 2019)		✓		✓	
Retroscope (Charapko, Ailijiang, Demirbas, & Kulkarni, 2019)					✓
LogAider (Di et al., 2018)		✓			
LogMine (Hamooni et al., 2016)		✓			
Craftsman (S. Zhang et al., 2020)		✓			
Drain (P. He, Zhu, Zheng, & Lyu, 2017)	✓				
Prilog (Tak et al., 2019)		✓			
Spell (M. Du & Li, 2019)	✓				
CRUDE (Gurumdimma, Jhumka, Liakata, Chuah, & Browne, 2016)			✓		
DeepAnt (Munir et al., 2019)			✓		
LogLens (Debnath et al., 2018)		✓	✓		
LogChain (Zhou et al., 2020)			✓		
HPCTOOLKIT (Adhianto et al., 2010)			✓	✓	✓
Log3C (S. He et al., 2018)		✓	✓		

These tools were developed for different purposes such as log parsing, log analysis, detection, prediction and recovery. This section will talk about 21 tools studied during the literature review based on the methodology used and its accuracy. This section also emphasizes the characteristics of every single tool in Table 6.

a. Log Preprocessing Tools:

LogAider tool (Di et al., 2018) establishes a temporal correlation between events to extract fatal events effectively. K-means clustering used for mining spatial correlations. Compared with failure records reported by admin, it shows 95% similarities.

A craftsman (S. Zhang et al., 2020) is a tool used for Syslog parsing, which is remarkably accurate, efficient in template matching and useful to various types of logs. It also enhances the computational efficiency by 6.88 to 10.25 times in template matching, and by 730 to 6847 times, it fails to find and merge similar templates to reduce the size.

Spell (M. Du & Li, 2019) is the event log parser working on the concept identification of semantic meaning for each field of log for understanding.

POP (P. He et al., 2018) (Hamooni et al., 2016) tool works on parallel processing on log data of BGL, HPC, HDFS, Zookeeper, Proxifier to reduce parsing time. In contrast, the log is processed by domain knowledge according to developers' simple regular expression rules.

The Drain (P. He et al., 2017), an online parsing tool, gives 99.9% accuracy on BGL, HDFS and Zookeeper data sets over LKE, IPLoM, SHISO and Spell parsers.

b. Anomaly or Failure Detection Tools:

CORRMEXT (Chuah et al., 2019) framework demonstrates the effectiveness of the concept of correlation between resource use data and message logs of the HPC system. CORRMEXT applies spearman rank and Pearson correlation algorithms (Chuah et al., 2011)(Chuah et al., 2018). Using this tool, one can generate error propagation paths if the failure occurs.

CRUDE (Gurumdimma et al., 2016) tool uses PCA unsupervised detection approach applied to event and resource usage log to find an odd job in distributed systems.

LogLens (Debnath et al., 2018) tool, the exemplary stateless algorithm, identifies the relationship between the log sequence of normal workflow execution and streaming logs and report anomalies. This approach performs 41x faster log parsing than the Logstash tool and saves up to 12096x person-hours in operational problem detection.

LogChain (Zhou et al., 2020) is a generalized tool that can apply to any cloud environment for failure detection in cloud management tasks. Where workflow labelled data is considered to compare with appropriate automata to identify the failure.

Log3C (S. He et al., 2018) is a tool available to locate impactful cloud system problems by correlating clusters of a log sequence and KPIs (Key performance indicators).

DeepAnt (Munir et al., 2019) tool uses the CNN approach to identify an anomaly in time series data. This tool is capable of detecting a small to a wide range of deviation in time series data.

c. Failure Prevention Tools:

Doomsday (Das et al., 2019) is the prediction tool for Cray systems that work on time-based phrases as a prediction mechanism. The authors claimed that the tool could notify failure in a node within 20 seconds to 2 min lead time. According to the research of (Fu et al., 2012), the event correlations graph (ECG) represents the correlation between the events, which is a prerequisite to designing association rules for event prediction using the Apriori LIS algorithm.

d. Failure Recovery Tools:

The research work (Charapko et al., 2019) proposed a Retroscope tool for retrospective monitoring of past consistent distributed snapshots, which can help in continuous monitoring of computer systems and recovery of data from failures or attack. HPCTOOLKIT (Adhianto et al., 2010) tool has been designed by focusing more on self-healing components.

7. Comprehensive Analysis of Existing Literature

This section will take an overview of significant points from the literature review on IT infrastructure monitoring. The analysis is targeting three components, such as the various infrastructures used, techniques and pinpointed limitations.

a. Type of Infrastructures used

Figure 8 presents the list of the infrastructures considered to handle the system failure problem in the studied literature.

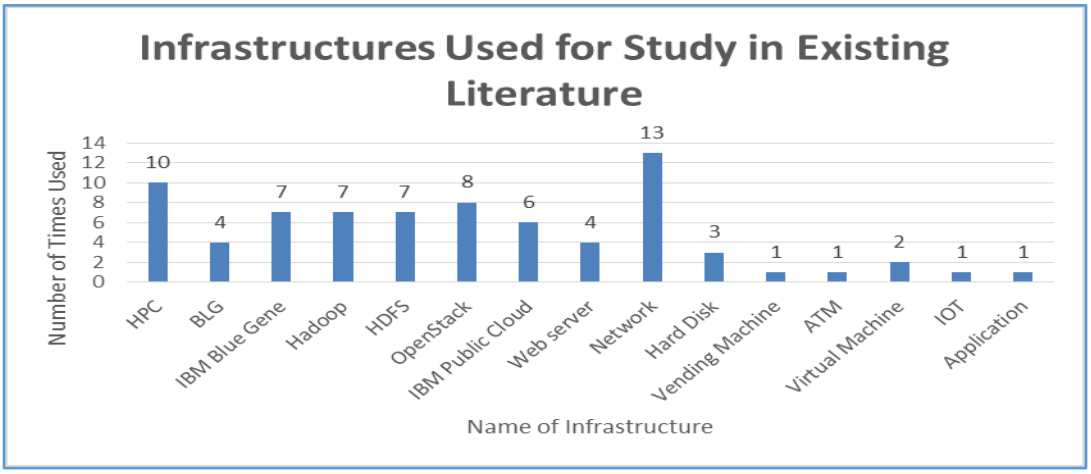

Figure 8 Infrastructures Used for Study in Existing Literature

We observed that the majority researcher has worked on supercomputers like HPC, BLG and IBM Blue Gene. A significant amount of work is also done in Hadoop and HDFS, followed by cloud systems such as OpenStack, IBM Public Cloud and the Webserver. A lot of work has been done in the identification and prevention of failures in the Network. Few researchers have focused on the hardware system to predict maintenance time and its health. Detection of node failure in a virtual machine, IoT is also one of the infrastructures explored by few researchers. Last but not least, a study has been done on software application. As the failure in software application can be the reason for computer system downtime.

b. Techniques used

Semantic analysis is a better choice than statistical analysis to derive the appropriate meaning from log data. Thus, many researchers have applied NLP techniques on log data considering log as normal text. Many researchers strongly use machine learning and deep learning techniques for anomaly or failure detection and prevention. A handful of researchers have explored autoencoder semi-supervised learning techniques. Making use of an autoencoder is helpful in case of substantial unlabeled log data.

c. Limitations in existing systems

1. Existing models in the literature are system-specific as each system is generating log in its own formats.
2. Log data are taken into account for analysis, assuming that the generated log is complete and accurate. But this assumption is not always valid.
3. Loss of important data may occur during log preprocessing due to data abstraction. In addition to that, sometimes encoded data is not in a readable format.
4. Existing models cannot detect every anomaly/failure in the system. The focus is only on the detection of significant anomalies/failures.
5. Available models can detect/identify failure but do not provide information (cause of failure, location or path, components involved) for taking necessary actions.
6. Experiments performed on dummy log data or real-time system data, but they have not considered sudden changes in the system's activity or spikes in log data.
7. Estimated time for prediction is not sufficient to take corrective actions. By the same token, the accuracy of prediction decreases with growing lead time.
8. Current systems are not getting updated dynamically, which cannot detect or predict anomalies/failure that has never appeared in history/ unreported. Furthermore, it cannot detect or predict anomalies/failure that occurs concurrently. Hence there is a need for a system that can handle such issues.

9. No fully automatic system is currently available for human intervention required in a previously unseen log sequence.
10. Root cause analysis is available only for past failures.

8. Conclusion

Downtime in any component of IT infrastructure ignites financial as well as productivity losses. Such system downtime is generally undesirable for continuously running applications. It is essential to maintain the IT infrastructures in the working state and reduce the downtime by early prediction of failure.

In concerning with the study done in the literature review, the first step in handling the failure is identifying the fault and finding the cause of failure. It is mandatory to know the system state, such as device status, error conditions, and other tasks, to take corrective actions. This information for analysis can be extracted from the system log data. The abnormal behaviors of the system can be identified by mining an enormous number of logs.

The literature review uncovers that many researchers have considered systems from different areas such as supercomputers, public cloud, servers, networks, application and hardware etc., despite that existing models are system-specific. Although current Models can detect the failure, they are not providing additional information like cause, location or path of failure. This information helps an administrator to take necessary corrective action and reduce the downtime quickly.

The study reveals that considerable work has been done in log preprocessing using natural language processing techniques. Research has followed three types of approaches for anomaly and failure detection such as rules base, correlation method, and classification. Many authors have used various machine learning and deep learning techniques for the sake of prediction. Researchers have focused their work on deep learning for prediction purposes, keeping data size and its ever-changing nature in mind.

Lack of early warning for failures is the predominant research gap identified during the literature review. For this reason, failure cannot be avoided due to a lack of time to take corrective action. Fully automated systems are not available even though the researcher has developed many solutions to detect and fix failures.

The study implies that failure rates are tremendous, even if much research has been done in this area. Thus, the future aspect of IT infrastructure monitoring demands research that can predict failure before it occurs. Furthermore, the literature study shows the essentiality of an automated system that can detect failures and perform self-correction actions in IT infrastructure to furnish the availability of components.

References

Adhianto, L., Banerjee, S., Fagan, M., Krentel, M., Marin, G., Mellor-Crummey, J., & Tallent, N. R. (2010). HPCTOOLKIT: Tools for performance analysis of optimized parallel programs. *Concurrency Computation Practice and Experience*, *22*(6), 685–701. https://doi.org/10.1002/cpe

Ahmad, S., Lavin, A., Purdy, S., & Agha, Z. (2017). Unsupervised real-time anomaly detection for streaming data. *Neurocomputing*, *262*, 134–147. https://doi.org/10.1016/j.neucom.2017.04.070

Amato, F., Cozzolino, G., Mazzeo, A., & Moscato, F. (2019). Detect and correlate information system events through verbose logging messages analysis. *Computing*, *101*(7), 819–830. https://doi.org/10.1007/s00607-018-0662-1

Aussel, N., Petetin, Y., & Chabridon, S. (2018). Improving performances of log mining for anomaly prediction through nlp-based log parsing. *Proceedings - 26th IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, MASCOTS 2018*, 237–243. https://doi.org/10.1109/MASCOTS.2018.00031

B, M. G. J., Cruzes, D. S., Angulo, J., & Fischer-h, S. (2016). for Cloud Customers. *International Conference on Cloud Computing and Services Science*, *1*, 38–57. https://doi.org/10.1007/978-3-319-29582-4

Basak, J., & Nagesh, P. C. (2016). A user-friendly log viewer for storage systems. *ACM Transactions on Storage*, *12*(3). https://doi.org/10.1145/2846101

Bertero, C., Roy, M., Sauvanaud, C., & Tredan, G. (2017). Experience Report: Log Mining Using Natural Language Processing and Application to Anomaly Detection. *Proceedings - International Symposium on Software Reliability Engineering, ISSRE*, *2017-Octob*, 351–360. https://doi.org/10.1109/ISSRE.2017.43

Borghesi, A., Bartolini, A., Lombardi, M., Milano, M., & Benini, L. (2019). A semisupervised autoencoder-based approach for anomaly detection in high performance computing systems. *Engineering Applications of Artificial Intelligence*, *85*(June), 634–644. https://doi.org/10.1016/j.engappai.2019.07.008

Bronevetsky, G., Laguna, I., De Supinski, B. R., & Bagchi, S. (2012). Automatic fault characterization via abnormality-enhanced classification. *Proceedings of the International Conference on Dependable Systems and Networks*. https://doi.org/10.1109/DSN.2012.6263926

Charapko, A., Ailijiang, A., Demirbas, M., & Kulkarni, S. (2019). Retroscope: Retrospective Monitoring of Distributed Systems. *IEEE Transactions on Parallel and Distributed Systems*, *30*(11), 2582–2594. https://doi.org/10.1109/TPDS.2019.2911944

Chaves, I. C., De Paula, M. R. P., Leite, L. G. M., Gomes, J. P. P., & MacHado, J. C. (2018). Hard

Disk Drive Failure Prediction Method Based on A Bayesian Network. *Proceedings of the International Joint Conference on Neural Networks*, *2018-July*, 1–7. https://doi.org/10.1109/IJCNN.2018.8489097

Chen, C., Singh, N., & Yajnik, S. (2012). Log analytics for dependable enterprise telephony. *Proceedings - 9th European Dependable Computing Conference, EDCC 2012*, 94–101. https://doi.org/10.1109/EDCC.2012.14

Choudhary, N., & Singh, P. (2013). Cloud Computing and Big Data Analytics. *International Journal of Engineering Research & Technology*, *2*(12), 2700–2704. https://doi.org/10.1007/978-3-319-28430-9

Chuah, E., Jhumka, A., Alt, S., Balouek-Thomert, D., Browne, J. C., & Parashar, M. (2019). Towards comprehensive dependability-driven resource use and message log-analysis for HPC systems diagnosis. *Journal of Parallel and Distributed Computing*, *132*, 95–112. https://doi.org/10.1016/j.jpdc.2019.05.013

Chuah, E., Jhumka, A., Alt, S., Damoulas, T., Gurumdimma, N., Sawley, M. C., … Browne, J. C. (2018). Enabling Dependability-Driven Resource Use and Message Log-Analysis for Cluster System Diagnosis. *Proceedings - 24th IEEE International Conference on High Performance Computing, HiPC 2017, 2017-Decem*, 317–327. https://doi.org/10.1109/HiPC.2017.00044

Chuah, E., Lee, G., Tjhi, W. C., Kuo, S. H., Hung, T., Hammond, J., … Browne, J. C. (2011). Establishing hypothesis for recurrent system failures from cluster log files. *Proceedings - IEEE 9th International Conference on Dependable, Autonomic and Secure Computing, DASC 2011*, 15–22. https://doi.org/10.1109/DASC.2011.27

Das, A., Mueller, F., Hargrove, P., Roman, E., & Baden, S. (2019). Doomsday: Predicting which node will fail when on supercomputers. *Proceedings - International Conference for High Performance Computing, Networking, Storage, and Analysis, SC 2018*, 108–121. https://doi.org/10.1109/SC.2018.00012

Debnath, B., Solaimani, M., Gulzar, M. A. G., Arora, N., Lumezanu, C., Xu, J., … Khan, L. (2018). LogLens: A real-time log analysis system. *Proceedings - International Conference on Distributed Computing Systems*, *2018-July*, 1052–1062. https://doi.org/10.1109/ICDCS.2018.00105

Di, S., Guo, H., Gupta, R., Pershey, E. R., Snir, M., & Cappello, F. (2018). Exploring Properties and Correlations of Fatal Events in a Large-Scale HPC System. *IEEE Transactions on Parallel and Distributed Systems*, *30*(2), 361–374. https://doi.org/10.1109/tpds.2018.2864184

Di, S., Guo, H., Pershey, E., Snir, M., & Cappello, F. (2019). Characterizing and Understanding HPC Job Failures over the 2K-Day Life of IBM BlueGene/Q System. *Proceedings - 49th*

Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN 2019, 473–484. https://doi.org/10.1109/DSN.2019.00055

Du, M., & Li, F. (2019). Spell: Online Streaming Parsing of Large Unstructured System Logs. *IEEE Transactions on Knowledge and Data Engineering*, *31*(11), 2213–2227. https://doi.org/10.1109/TKDE.2018.2875442

Du, M., Li, F., Zheng, G., & Srikumar, V. (2017). DeepLog: Anomaly detection and diagnosis from system logs through deep learning. *Proceedings of the ACM Conference on Computer and Communications Security*, 1285–1298. https://doi.org/10.1145/3133956.3134015

Du, S., & Cao, J. (2015). Behavioral anomaly detection approach based on log monitoring. *2015 International Conference on Behavioral, Economic and Socio-Cultural Computing, BESC 2015*, (Besc), 188–194. https://doi.org/10.1109/BESC.2015.7365981

Dua, K., Choudhury, T., Rajanikanth, U., & Choudhury, A. (2019). CGI based syslog management system for virtual machines. *Spatial Information Research*. https://doi.org/10.1007/s41324-019-00308-7

El-Masri, D., Petrillo, F., Guéhéneuc, Y. G., Hamou-Lhadj, A., & Bouziane, A. (2020). A systematic literature review on automated log abstraction techniques. *Information and Software Technology*, *122*(December 2018), 106276. https://doi.org/10.1016/j.infsof.2020.106276

Farshchi, M., Schneider, J. G., Weber, I., & Grundy, J. (2018). Metric selection and anomaly detection for cloud operations using log and metric correlation analysis. *Journal of Systems and Software*, *137*, 531–549. https://doi.org/10.1016/j.jss.2017.03.012

Fu, X., Ren, R., Mckee, S. A., Zhan, J., & Sun, N. (2014). Digging deeper into cluster system logs for failure prediction and root cause diagnosis. *2014 IEEE International Conference on Cluster Computing, CLUSTER 2014*, (2), 103–112. https://doi.org/10.1109/CLUSTER.2014.6968768

Fu, X., Ren, R., Zhan, J., Zhou, W., Jia, Z., & Lu, G. (2012). Logmaster: Mining event correlations in logs of large-scale cluster systems. *Proceedings of the IEEE Symposium on Reliable Distributed Systems*, 71–80. https://doi.org/10.1109/SRDS.2012.40

Gainaru, A., Cappello, F., Fullop, J., Trausan-Matu, S., & Kramer, W. (2011). Adaptive event prediction strategy with dynamic time window for large-scale HPC systems. *Managing Large-Scale Systems via the Analysis of System Logs and the Application of Machine Learning Techniques, SLAML'11*, 1–8. https://doi.org/10.1145/2038633.2038637

Gainaru, A., Cappello, F., Snir, M., & Kramer, W. (2012). Fault prediction under the microscope: A closer look into HPC systems. *International Conference for High Performance Computing, Networking, Storage and Analysis, SC*. https://doi.org/10.1109/SC.2012.57

Gainaru, A., Cappello, F., Snir, M., & Kramer, W. (2013). Failure prediction for HPC systems and applications: Current situation and open issues. *International Journal of High Performance Computing Applications*, *27*(3), 273–282. https://doi.org/10.1177/1094342013488258

Gao, X., Zha, S., Li, X., Yan, B., Jing, X., Li, J., & Xu, J. (2019). Incremental Prediction Model of Disk Failures Based on the Density Metric of Edge Samples. *IEEE Access*, *7*, 114285–114296. https://doi.org/10.1109/access.2019.2935628

Ghiasvand, S. (2019). UPAD: Unsupervised privacy-aware anomaly detection in high performance computing systems. *ICPRAM 2019 - Proceedings of the 8th International Conference on Pattern Recognition Applications and Methods*, (Icpram), 852–859. https://doi.org/10.5220/0007582208520859

Gurumdimma, N., Jhumka, A., Liakata, M., Chuah, E., & Browne, J. (2016). CRUDE: Combining Resource Usage Data and Error Logs for Accurate Error Detection in Large-Scale Distributed Systems. *Proceedings of the IEEE Symposium on Reliable Distributed Systems*, 51–60. https://doi.org/10.1109/SRDS.2016.017

Hamooni, H., Debnath, B., Xu, J., Zhang, H., Jiang, G., & Mueen, A. (2016). LogMine: Fast pattern recognition for log analytics. *International Conference on Information and Knowledge Management, Proceedings*, *24-28-Octo*(May 2018), 1573–1582. https://doi.org/10.1145/2983323.2983358

Hassani, M., Shang, W., Shihab, E., & Tsantalis, N. (2018). Studying and detecting log-related issues. In *Empirical Software Engineering* (Vol. 23). https://doi.org/10.1007/s10664-018-9603-z

He, P., Zhu, J., He, S., Li, J., & Lyu, M. R. (2018). Towards Automated Log Parsing for Large-Scale Log Data Analysis. *IEEE Transactions on Dependable and Secure Computing*, *15*(6), 931–944. https://doi.org/10.1109/TDSC.2017.2762673

He, P., Zhu, J., Zheng, Z., & Lyu, M. R. (2017). Drain: An Online Log Parsing Approach with Fixed Depth Tree. *Proceedings - 2017 IEEE 24th International Conference on Web Services, ICWS 2017*, 33–40. https://doi.org/10.1109/ICWS.2017.13

He, S., Lin, Q., Lou, J. G., Zhang, H., Lyu, M. R., & Zhang, D. (2018). Identifying impactful service system problems via log analysis. *ESEC/FSE 2018 - Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 60–70. https://doi.org/10.1145/3236024.3236083

Huang, L., Ke, X., Wong, K., & Mankovskii, S. (2010). Symptom-based problem determination using log data abstraction. *Proceedings of the 2010 Conference of the Center for Advanced Studies on Collaborative Research, CASCON'10*, 313–326.

https://doi.org/10.1145/1923947.1923979

IT service management (ITSM): process, benefits, ITSM vs ITIL, best practices & metrics. (n.d.). Retrieved April 30, 2020, from https://www.manageengine.com/products/service-desk/itsm/

Jain, S., Singh, I., Chandra, A., Zhang, Z. L., & Bronevetsky, G. (2009). Extracting the textual and temporal structure of supercomputing logs. *16th International Conference on High Performance Computing, HiPC 2009 - Proceedings*, 254–263. https://doi.org/10.1109/HIPC.2009.5433202

Jha, S., Formicola, V., Di Martino, C., Dalton, M., Kramer, W. T., Kalbarczyk, Z., & Iyer, R. K. (2018). Resiliency of HPC interconnects: A case study of interconnect failures and recovery in blue waters. *IEEE Transactions on Dependable and Secure Computing*, *15*(6), 915–930. https://doi.org/10.1109/TDSC.2017.2737537

Jia, T., Chen, P., Yang, L., Li, Y., Meng, F., & Xu, J. (2017). An Approach for Anomaly Diagnosis Based on Hybrid Graph Model with Logs for Distributed Services. *Proceedings - 2017 IEEE 24th International Conference on Web Services, ICWS 2017*, 25–32. https://doi.org/10.1109/ICWS.2017.12

Jia, T., Yang, L., Chen, P., Li, Y., Meng, F., & Xu, J. (2017). LogSed: Anomaly Diagnosis through Mining Time-Weighted Control Flow Graph in Logs. *IEEE International Conference on Cloud Computing, CLOUD, 2017-June*, 447–455. https://doi.org/10.1109/CLOUD.2017.64

Kimura, T., Watanabe, A., Toyono, T., & Ishibashi, K. (2019). Proactive failure detection learning generation patterns of large-scale network logs. *IEICE Transactions on Communications*, (2), 306–316. https://doi.org/10.1587/transcom.2018EBP3103

Kobayashi, S., Otomo, K., Fukuda, K., & Esaki, H. (2018). Mining Causality of Network Events in Log Data. *IEEE Transactions on Network and Service Management*, *15*(1), 53–67. https://doi.org/10.1109/TNSM.2017.2778096

Konno, S., & Défago, X. (2019). Approximate QoS rule derivation based on root cause analysis for cloud computing. *Proceedings of IEEE Pacific Rim International Symposium on Dependable Computing, PRDC, 2019-Decem*, 33–42. https://doi.org/10.1109/PRDC47002.2019.00020

Le, D. C., & Zincir-Heywood, N. (2020). A Frontier: Dependable, Reliable and Secure Machine Learning for Network/System Management. *Journal of Network and Systems Management*, (0123456789). https://doi.org/10.1007/s10922-020-09512-5

Li, Y., Jiang, Z. M. J., Li, H., Hassan, A. E., He, C., Huang, R., … Chen, P. (2020). Predicting Node Failures in an Ultra-Large-Scale Cloud Computing Platform. *ACM Transactions on Software Engineering and Methodology*, *29*(2). https://doi.org/10.1145/3385187

Li, Z., Davidson, M., Fu, S., Blanchard, S., & Lang, M. (2018). Converting unstructured system

logs into structured event list for anomaly detection. *ACM International Conference Proceeding Series*. https://doi.org/10.1145/3230833.3230855

Li, Z., Davidson, M., Fu, S., Blanchard, S., & Lang, M. (2019). Event Block Identification and Analysis for Effective Anomaly Detection to Build Reliable HPC Systems. *Proceedings - 20th International Conference on High Performance Computing and Communications, 16th International Conference on Smart City and 4th International Conference on Data Science and Systems, HPCC/SmartCity/DSS 2018*, 781–788. https://doi.org/10.1109/HPCC/SmartCity/DSS.2018.00132

Lin, Q., Zhang, H., Lou, J. G., Zhang, Y., & Chen, X. (2016). Log clustering based problem identification for online service systems. *Proceedings - International Conference on Software Engineering*, 102–111. https://doi.org/10.1145/2889160.2889232

Liu, Y., Lv, J., Ma, S., & Yao, W. (2018). The runtime system problem identification method based on log analysis. *Proceedings - International Conference on Computer Communications and Networks, ICCCN*, *2018-July*(61300007), 1–7. https://doi.org/10.1109/ICCCN.2018.8487466

Lu, S., Rao, B. B., Wei, X., Tak, B., Wang, L., & Wang, L. (2017). Log-based Abnormal Task Detection and Root Cause Analysis for Spark. *Proceedings - 2017 IEEE 24th International Conference on Web Services, ICWS 2017*, 389–396. https://doi.org/10.1109/ICWS.2017.135

Lu, S., Wei, X., Li, Y., & Wang, L. (2018). Detecting anomaly in big data system logs using convolutional neural network. *Proceedings - IEEE 16th International Conference on Dependable, Autonomic and Secure Computing, IEEE 16th International Conference on Pervasive Intelligence and Computing, IEEE 4th International Conference on Big Data Intelligence and Computing and IEEE 3*, 159–165. https://doi.org/10.1109/DASC/PiCom/DataCom/CyberSciTec.2018.00037

Meng, W., Liu, Y., Zhang, S., Pei, D., Dong, H., Song, L., & Luo, X. (2019). Device-Agnostic Log Anomaly Classification with Partial Labels. *2018 IEEE/ACM 26th International Symposium on Quality of Service, IWQoS 2018*, (1), 1–6. https://doi.org/10.1109/IWQoS.2018.8624141

Meng, W., Liu, Y., Zhu, Y., Zhang, S., Pei, D., Liu, Y., … Zhou, R. (2019). Loganomaly: Unsupervised detection of sequential and quantitative anomalies in unstructured logs. *IJCAI International Joint Conference on Artificial Intelligence*, *2019-Augus*, 4739–4745. https://doi.org/10.24963/ijcai.2019/658

Munir, M., Siddiqui, S. A., Dengel, A., & Ahmed, S. (2019). DeepAnT: A Deep Learning Approach for Unsupervised Anomaly Detection in Time Series. *IEEE Access*, *7*, 1991–2005. https://doi.org/10.1109/ACCESS.2018.2886457

Nandi, A., Mandal, A., Atreja, S., Dasgupta, G. B., & Bhattacharya, S. (2016). Anomaly detection

using program control flow graph mining from execution logs. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, *13-17-Augu*, 215–224. https://doi.org/10.1145/2939672.2939712

Oliner, A., & Stearley, J. (2007). What supercomputers say: A study of five system logs. *Proceedings of the International Conference on Dependable Systems and Networks*, 575–584. https://doi.org/10.1109/DSN.2007.103

Otomo, K., Kobayashi, S., Fukuda, K., & Esaki, H. (2019). Latent variable based anomaly detection in network system logs. *IEICE Transactions on Information and Systems*, *E102D*(9), 1644–1652. https://doi.org/10.1587/transinf.2018OFP0007

Ozcelik, B., & Yilmaz, C. (2016). Seer: A Lightweight Online Failure Prediction Approach. *IEEE Transactions on Software Engineering*, *42*(1), 26–46. https://doi.org/10.1109/TSE.2015.2442577

Pal, A., & Kumar, M. (2019). DLME: Distributed Log Mining Using Ensemble Learning for Fault Prediction. *IEEE Systems Journal*, *13*(4), 3639–3650. https://doi.org/10.1109/JSYST.2019.2904513

Pecchia, A., Weber, I., Cinque, M., & Ma, Y. (2020). Discovering process models for the analysis of application failures under uncertainty of event logs. *Knowledge-Based Systems*, *189*(xxxx). https://doi.org/10.1016/j.knosys.2019.105054

Pettinato, M., Gil, J. P., Galeas, P., & Russo, B. (2019). Log mining to re-construct system behavior: An exploratory study on a large telescope system. *Information and Software Technology*, *114*(June), 121–136. https://doi.org/10.1016/j.infsof.2019.06.011

Pitakrat, T., Grunert, J., Kabierschke, O., Keller, F., & Van Hoorn, A. (2014). A framework for system event classification and prediction by means of machine learning. *Proceedings of the 8th International Conference on Performance Evaluation Methodologies and Tools, VALUETOOLS 2014*, *8*, 173–180. https://doi.org/10.4108/icst.valuetools.2014.258197

Pitakrat, T., Okanović, D., van Hoorn, A., & Grunske, L. (2018). Hora: Architecture-aware online failure prediction. *Journal of Systems and Software*, *137*, 669–685. https://doi.org/10.1016/j.jss.2017.02.041

Qi, G., Tsai, W. T., Li, W., Zhu, Z., & Luo, Y. (2017). A cloud-based triage log analysis and recovery framework. *Simulation Modelling Practice and Theory*, *77*, 292–316. https://doi.org/10.1016/j.simpat.2017.07.003

Rawat, A., Sushil, R., Agarwal, A., & Sikander, A. (2018). A New Approach for VM Failure Prediction using Stochastic Model in Cloud. *IETE Journal of Research*, *0*(0), 1–8. https://doi.org/10.1080/03772063.2018.1537814

Ren, R., Cheng, J., Yin, Y., Zhan, J., Wang, L., Li, J., & Luo, C. (2019). Deep Convolutional Neural Networks for Log Event Classification on Distributed Cluster Systems. *Proceedings - 2018 IEEE International Conference on Big Data, Big Data 2018*, 1639–1646. https://doi.org/10.1109/BigData.2018.8622611

Ren, Y., Gu, Z., Wang, Z., Tian, Z., Liu, C., Lu, H., … Guizani, M. (2020). System log detection model based on conformal prediction. *Electronics (Switzerland)*, *9*(2). https://doi.org/10.3390/electronics9020232

Root Cause Analysis (RCA) for IT – BMC Blogs. (n.d.). Retrieved April 30, 2020, from https://www.bmc.com/blogs/root-cause-analysis/

Roumani, Y., & Nwankpa, J. K. (2019). An empirical study on predicting cloud incidents. *International Journal of Information Management*, *47*(January), 131–139. https://doi.org/10.1016/j.ijinfomgt.2019.01.014

Saadatfar, H., Fadishei, H., & Deldari, H. (2012). Predicting job failures in auvergrid based on workload log analysis. *New Generation Computing*, *30*(1), 73–94. https://doi.org/10.1007/s00354-012-0105-z

Shen, J., Wan, J., Lim, S. J., & Yu, L. (2018). Random-forest-based failure prediction for hard disk drives. *International Journal of Distributed Sensor Networks*, *14*(11). https://doi.org/10.1177/1550147718806480

Sorkunlu, N., Anh Luong, D. T., & Chandola, V. (2019). DynamicMF: A Matrix Factorization Approach to Monitor Resource Usage in High Performance Computing Systems. *Proceedings - 2018 IEEE International Conference on Big Data, Big Data 2018*, 1302–1307. https://doi.org/10.1109/BigData.2018.8622425

Tak, B., Park, S., & Kudva, P. (2019). Priolog: Mining important logs via temporal analysis and prioritization. *Sustainability (Switzerland)*, *11*(22), 1–17. https://doi.org/10.3390/su11226306

Tan, Y., & Gu, X. (2010). On predictability of system anomalies in real world. *Proceedings - 18th Annual IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, MASCOTS 2010*, 133–140. https://doi.org/10.1109/MASCOTS.2010.22

The Cost of Downtime - Andrew Lerner. (n.d.). Retrieved April 30, 2020, from https://blogs.gartner.com/andrew-lerner/2014/07/16/the-cost-of-downtime/

Tiwari, D., Gupta, S., & Vazhkudai, S. S. (2014). Lazy checkpointing: Exploiting temporal locality in failures to mitigate checkpointing overheads on extreme-scale systems. *Proceedings - 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN 2014*, 25–36. https://doi.org/10.1109/DSN.2014.101

Wang, J., Li, C., Han, S., Sarkar, S., & Zhou, X. (2017). Predictive maintenance based on event-log analysis: A case study. *IBM Journal of Research and Development*, *61*(1), 121–132. https://doi.org/10.1147/JRD.2017.2648298

Wang, Jin, Tang, Y., He, S., Zhao, C., Sharma, P. K., Alfarraj, O., & Tolba, A. (2020). LogEvent2vec: LogEvent-to-vector based anomaly detection for large-scale logs in internet of things. *Sensors (Switzerland)*, *20*(9), 1–19. https://doi.org/10.3390/s20092451

Wang, M., Xu, L., & Guo, L. (2018). Anomaly detection of system logs based on natural language processing and deep learning. *2018 4th International Conference on Frontiers of Signal Processing, ICFSP 2018*, 140–144. https://doi.org/10.1109/ICFSP.2018.8552075

Wang, X., Wang, D., Zhang, Y., Jin, L., & Song, M. (2019). Unsupervised learning for log data analysis based on behavior and attribute features. *ACM International Conference Proceeding Series*, 510–518. https://doi.org/10.1145/3349341.3349460

Weng, J., Wang, J. H., Yang, J., & Yang, Y. (2018). Root Cause Analysis of Anomalies of Multitier Services in Public Clouds. *IEEE/ACM Transactions on Networking*, *26*(4), 1646–1659. https://doi.org/10.1109/TNET.2018.2843805

Wu, P., Lu, Z., Zhou, Q., Lei, Z., Li, X., Qiu, M., & Hung, P. C. K. (2019). Bigdata logs analysis based on seq2seq networks for cognitive Internet of Things. *Future Generation Computer Systems*, *90*, 477–488. https://doi.org/10.1016/j.future.2018.08.021

Xiang, S., Huang, D., & Li, X. (2019). A Generalized Predictive Framework for Data Driven Prognostics and Diagnostics using Machine Logs. *IEEE Region 10 Annual International Conference, Proceedings/TENCON*, *2018-Octob*(October), 695–700. https://doi.org/10.1109/TENCON.2018.8650152

Xie, X., Jin, Z., Wang, J., Yang, L., Lu, Y., & Li, T. (2020). Confidence guided anomaly detection model for anti-concept drift in dynamic logs. *Journal of Network and Computer Applications*, *162*(February), 1–10. https://doi.org/10.1016/j.jnca.2020.102659

Yoo, W., Sim, A., & Wu, K. (2016). Machine learning based job status prediction in scientific clusters. *Proceedings of 2016 SAI Computing Conference, SAI 2016*, 44–53. https://doi.org/10.1109/SAI.2016.7555961

Yuan, W., Lu, S., Sun, H., & Liu, X. (2020). How are distributed bugs diagnosed and fixed through system logs? *Information and Software Technology*, *119*(June 2019), 106234. https://doi.org/10.1016/j.infsof.2019.106234

Yuan, Y., Anu, H., Shi, W., Liang, B., & Qin, B. (2019). Learning-based anomaly cause tracing with synthetic analysis of logs from multiple cloud service components. *Proceedings - International Computer Software and Applications Conference*, *1*, 66–71.

https://doi.org/10.1109/COMPSAC.2019.00019

Yuan, Y., Shi, W., Liang, B., & Qin, B. (2019). An approach to cloud execution failure diagnosis based on exception logs in openstack. *IEEE International Conference on Cloud Computing, CLOUD*, *2019-July*, 124–131. https://doi.org/10.1109/CLOUD.2019.00031

Zhang, K., Xu, J., Min, M. R., Jiang, G., Pelechrinis, K., & Zhang, H. (2016). Automated IT system failure prediction: A deep learning approach. *Proceedings - 2016 IEEE International Conference on Big Data, Big Data 2016*, 1291–1300. https://doi.org/10.1109/BigData.2016.7840733

Zhang, S., Song, L., Zhang, M., Liu, Y., Meng, W., Bu, J., … Zhang, Y. (2020). Efficient and Robust Syslog Parsing for Network Devices in Datacenter Networks. *IEEE Access*, *8*, 30245–30261. https://doi.org/10.1109/ACCESS.2020.2972691

Zhang, X., Xu, Y., Lin, Q., Qiao, B., Zhang, H., Dang, Y., … Zhang, D. (2019). Robust log-based anomaly detection on unstable log data. *ESEC/FSE 2019 - Proceedings of the 2019 27th ACM Joint Meeting European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 807–817. https://doi.org/10.1145/3338906.3338931

Zheng, Z., Lan, Z., Gupta, R., Coghlan, S., & Beckman, P. (2010). A practical failure prediction with location and lead time for Blue Gene/P. *Proceedings of the International Conference on Dependable Systems and Networks*, 15–22. https://doi.org/10.1109/DSNW.2010.5542627

Zhou, P., Wang, Y., Li, Z., Tyson, G., Guan, H., & Xie, G. (2020). Logchain: Cloud workflow reconstruction & troubleshooting with unstructured logs. *Computer Networks*, *175*(March). https://doi.org/10.1016/j.comnet.2020.107279

Zou, D. Q., Qin, H., & Jin, H. (2016). UiLog: Improving Log-Based Fault Diagnosis by Log Analysis. *Journal of Computer Science and Technology*, *31*(5), 1038–1052. https://doi.org/10.1007/s11390-016-1678-7