Failure Prediction by Utilizing Log Analysis: A Systematic Mapping Study

Dipta Das
dipta_das1@baylor.edu
Computer Science, Baylor University
Waco, Texas, USA

Micah Schiewe
micah_schiewe1@baylor.edu
Computer Science, Baylor University
Waco, Texas, USA

Elizabeth Brighton
elizabeth_brighton@baylor.edu
Computer Science, Baylor University
Waco, Texas, USA

Mark Fuller
mark_fuller1@baylor.edu
Computer Science, Baylor University
Waco, Texas, USA

Tomas Cerny
tomas_cerny@baylor.edu
Computer Science, Baylor University
Waco, Texas, USA

Miroslav Bures
miroslav.bures@fel.cvut.cz
CS, FEE, Czech Technical University
Prague, Czech Republic

Karel Frajtak
frajtak@fel.cvut.cz
CS, FEE, Czech Technical University
Prague, Czech Republic

Dongwan Shin
dongwan.shin@nmt.edu
Computer Science, New Mexico Tech
Socorro, New Mexico, USA

Pavel Tisnovsky
ptisnovs@redhat.com
Red Hat
Brno, Czech Republic

ABSTRACT

In modern computing, log files provide a wealth of information regarding the past of a system, including the system failures and security breaches that cost companies and developers a fortune in both time and money. While this information can be used to attempt to recover from a problem, such an approach merely mitigates the damage that has already been done. Detecting problems, however, is not the only information that can be gathered from log files. It is common knowledge that segments of log files, if analyzed correctly, can yield a good idea of what the system is likely going to do next in real-time, allowing a system to take corrective action before any negative actions occur. In this paper, the authors put forth a systematic map of this field of log prediction, screening several hundred papers and finally narrowing down the field to approximately 30 relevant papers. These papers, when broken down, give a good idea of the state of the art, methodologies employed, and future challenges that still must be overcome. Findings and conclusions of this study can be applied to a variety of software systems and components, including classical software systems, as well as software parts of control, or the Internet of Things (IoT) systems.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; • **Computer systems organization** → *Reliability*; *Availability*; *Maintainability and maintenance.*

RACS '20, October 13–16, 2020, Gwangju, Republic of Korea
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-8025-6/20/10…$15.00
https://doi.org/10.1145/3400286.3418263

KEYWORDS

Failure Prediction, Defect Prediction, Error Logs, Log Analysis, Mapping Study

ACM Reference Format:
Dipta Das, Micah Schiewe, Elizabeth Brighton, Mark Fuller, Tomas Cerny, Miroslav Bures, Karel Frajtak, Dongwan Shin, and Pavel Tisnovsky. 2020. Failure Prediction by Utilizing Log Analysis: A Systematic Mapping Study. In *International Conference on Research in Adaptive and Convergent Systems (RACS '20), October 13–16, 2020, Gwangju, Republic of Korea.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3400286.3418263

1 INTRODUCTION

Traditionally log files are used in order to trace back errors after they have occurred. While this can be beneficial, it often involves hours of programmers time to trace back through the logs and find the root issue. The field of error detection has been widely studied [24, 30] as a response to the manual time wasted; however, these solutions are passive in nature and do not give the needed resources to react to the errors when they occur. Thus, the natural next step is error prediction, where the error can be seen in advance and dealt with accordingly [24, 30]. There are many practical applications for error prediction, including responding to network attacks [20, 30], avoiding hardware failure [17, 23], and reducing manual time spent scanning logs for potential errors.

The purpose of this paper is to categorize the current methods of log prediction discovered in this field of research. This includes analyzing everything from the specific type of logs used in each relevant paper discovered to discussing the relative success and failure of each algorithm implemented for currently running log files. The authors of this study have found that many obstacles are in the way of efficient prediction methods, including conforming to individual log structure and scaling to large-scale systems. There are also many breakthroughs in this field including enhancing popular algorithms or creating novel machine learning methods of error prediction.

Findings and conclusions of this study are relevant for a variety of software components or systems, e.g., enterprise software systems, software parts of control systems, or software components of the Internet of Things (IoT) systems, where security vulnerabilities are often reported as a significant problem [1, 2].

The rest of the paper is organized as follows. Section 2 gives a general background on log file, error detection and failure prediction. Section 3 describes how the authors collected and analyzed the relevant papers on log prediction. Section 4 answers the research questions listed in section 3. And finally, we concluded the paper in section 5 with a general summary of our contributions along with future works.

2 BACKGROUND

In general, log files record the state of a system at any given time [37]. Such files usually record information regarding a network, system, or merely a flow of events, and in turn, provide a wealth of information regarding failures [12, 30], which can be used to diagnose problems. However, as systems have grown larger, so too have their logs. This growth has long since reached the point of becoming bloated beyond any human's ability to manually parse–thus forcing developers to resort to automated methods [37]. Since logs do not adhere to any standard format [22, 28], though, such automated parsing can be close to impossible without extreme customization. This issue is compounded twofold: first, by the number of issues recorded in logs being dwarfed by the total number of log entries [39], and second, by the tendency of software systems to change logging behavior and patterns over time [38]. Generally, this three-way mystery has presented too much volume and too complex a structure for a standard algorithm to be applied to, forcing developers to fall back on Machine Learning (ML) as a solution. This particular topic, however, will be expanded upon further when discussing the findings of this mapping study.

Automatic detection of errors from log files is not enough, however; after all, since logs record the behavior of a system, a logged error is one which has already occurred and thus done damage. Traditionally, handling such errors is done with less thought put towards logs. Instead, systems are designed as *'fault-tolerant'*, expending effort towards periodic checkpointing of the full system to minimize work lost in the event of failures of any part [17]. This, however, wastes a great deal of CPU time checkpointing components that are nowhere near failure, and can still lose progress on components that fail just before the system is checkpointed [17]. Due to these and other issues, research has begun moving in the direction of attempting to predict failures before they happen, allowing the minimization or elimination of damage before it occurs [24, 30].

Thanks to the fact that predicting from logs is such an integral discipline which will only grow as systems grow larger, it is important to document the current state of the art to clarify where research is headed and what problems remain to be addressed. In short, a systematic mapping study is in order. There is already at least one such study available, "Outcome-Oriented Predictive Process Monitoring: Review and Benchmark" by Teinemaa et. all [34]. However, this study may not suffice for answering the questions raised here. The study approaches the problem from the angle of classifying the outcome of business cases, which narrows the focus to exclude applications (such as software fault prediction) and thus does not encompass the scope of a study that is needed. This question is reinforced by the conclusion citing having found 14 works, a far smaller number than our search yielded [34]. One other similar study, "The impact of feature reduction techniques on defect prediction models" by Kondo et. all, also seems superficially similar [21]. However, this research focused primarily on how feature reduction improved defect prediction, not on the methodology used in this prediction [21].

3 MAPPING STUDY METHOD

In this study, we carried out a structured procedure to accumulate and synthesize the research works on failure prediction in computing systems. To find existing groundwork relevant to the specific field of interest, we followed a software engineering approach of systematic mapping studies [27].

In the first phase of our mapping study, we defined a set of research questions and refine them over time such that answers to those questions represent a comprehensive understanding of the topic under consideration: failure prediction using log analysis. Next, we identified the search terms for querying across different indexing sites and portals. This phase required a trial-and-error process to finalize the search terms relevant to our topic. Once all papers are gathered, we manually filtered out out-of-scope papers by reading through the abstracts and prepared a shortlist. Finally, we rigorously analyzed the shortlisted papers to answer the research questions.

The questions we examined in this mapping study are as follows:

- RQ1 What is the trend in research over time? Has more or less been done recently?
- RQ2 Which countries and people are most active in this field of research?
- RQ3 What kinds of logs are being analyzed?
- RQ4 What is being predicted from those logs?
- RQ5 What limitations and challenges face the field of log prediction?
- RQ6 What methodology, algorithms, tests, and results were garnered by the study?
- RQ7 What are the future directions for failure prediction?

We used four indexing sites and portals (indexers), including IEEE Xplore, ACM Digital Library (DL), ScienceDirect, and SpringerLink. We tailored our search queries to look for papers related to failure prediction using log analysis. We divided our search query into four parts. In the first part of our query, we included the search term *"log analysis"*, however, we also considered related terms like *"log mining"* and *"log extraction"*. In the second part, we used similar terms that represent failures: *error, issue, defect, fault, failure and crash*. For the last part, we injected the terms *"prediction"* or, *"forecast"* to refine our search results. The full search query is presented in the Listing 1.

Each paper after the initial filtering was then manually evaluated on its title and abstract to determine if it was a fit for the scope we are looking for. Our search query returned a large number of papers; however, after the manual processing, we found that most of them are related to failure detection instead of failure prediction. Apart

Listing 1: The Search Query for the Indexers

```
(log analysis OR log mining OR log extraction)
   AND (error OR fault OR issue OR defect OR failure OR crash)
   AND (prediction OR forecast)
```

Table 1: Search Query Results for Various Index Sites

Indexer	Search Results	Filtered	Referenced	Total Relevant
ACM DL	9	1	6	7
IEEE Xplore	123	7	5	12
SpringerLink	172	3	1	4
ScienceDirect	5[1]	1	3	4
Others	-	-	3	3
Total	309	12	18	30

from that, we have also discarded papers that are in non-English languages, papers without available full-text, opinion-based papers, and short papers with less than four pages. Then we went through the related work section of the remaining papers to include relevant studies that the search query omitted.

The results of our search queries and manual filtering are listed in Table 1 along with the papers we found by exploring related work sections. Once we narrowed down the relevant works to about 30 papers, we thoroughly studied them to discover current trends in failure predictions based on log analysis along with the challenges and solutions they presented.

4 ANALYSIS RESULTS

Out of 309 papers returned by the search, we recognized a very small number of relevant works. The majority of the works focused on anomaly detection from log files in general instead of predicting failures. Only 30 papers are considered for the final analysis, including the ones with only a partial match. In this section, we present the findings of the study by answering the research questions in separate subsections. The validity of our systematic study is discussed in the last subsection.

4.1 Trends

The idea of log prediction has been around since the 1990s. Even though research started in the 1990s, not much research was done until around 2008 when a larger emphasis was placed on error detection. This emphasis occurred naturally as a reaction to the ever-growing size and complexity of software systems and the escalating damage that system failure creates. Figure 1 illustrates all of the papers the authors included in this study. As can be seen from the graph above, the general trend of research on log prediction has been increasing over time, especially over the last few years. Out of approximately 30 papers, the authors choose for this paper, Figure 2 shows an even and steady increase in the number of papers written.

[1]Since ScienceDirect limits the number of boolean connectors we split second part of our query into two separate queries: (issue OR crash OR failure) and (defect OR fault OR error)

4.2 Countries and People

There were no common names found in the research papers the authors analyzed. In general, each person contributed to one or two papers, with only one person contributing to three research papers. This shows us that the research is fairly widespread, and not contained to one group of people. This conclusion can also be seen in Figure 3, which outlines the number of people in each country that are involved in researching error prediction through log analysis. Out of the 30 papers analyzed, the United States currently has the most people involved in this research. The other countries that contributed the most in the research of log prediction are China and Italy.

4.3 Types and Sources of Logs

Regarding the research question RQ3, the types of logs being analyzed, of the selected papers, many of them address the content of the log data analyzed. These logs are all taken from existing systems with either in real-time or deferred analysis techniques. Out of the group of papers examined, four of the papers focused on the general application of an algorithm to any log type [10, 34, 34, 37]. Five focused on logs of scheduled processes or process meta-data, such as distributed computing [17, 19, 25, 29, 32]. Four were solely dedicated to network communication or IoT for the purpose of security [13, 20, 22, 35], while the rest were solely dedicated to the prediction and avoidance of errors within the system.

4.4 Pre-Processing

Format and content is crucial to the efficient analysis of logs. Of the papers analyzed, 17 [10–12, 16, 18–22, 26, 32, 34, 35, 37–39, 41] directly mentioned a pre-processing step. In six of these papers, this crucial step was used to remove redundant or irrelevant data

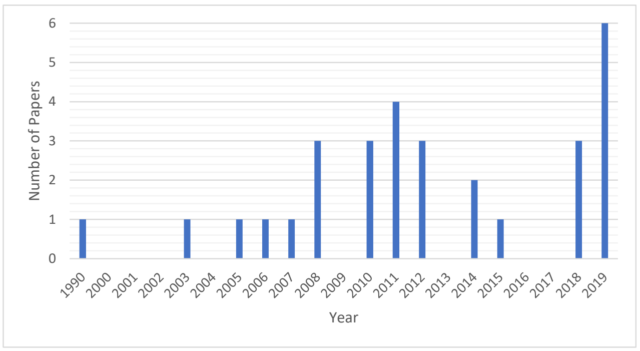

Figure 1: Number of papers found per year

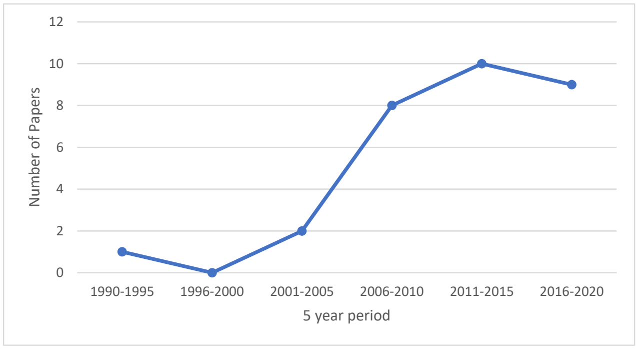

Figure 2: Number of papers found per 5 years

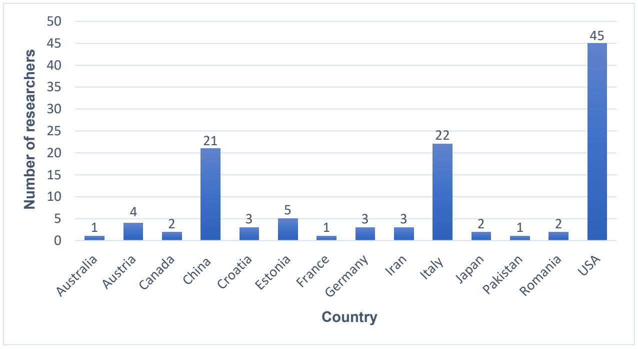

Figure 3: Number of researchers involved per country

[16, 18, 20, 32, 38, 41]. This redundant or irrelevant data came in the form of duplicate log entries or log entries that are not pertinent to the analysis. Five used this step to generally reformat the data into a more structured form [10, 18, 26, 32, 34], this new form decreases runtime and improves overall analysis. The inconsistency of time stamps and the aggregation of distributed logs is a domain of increasing size as distributed systems become more common. In microservices, distributed systems, Internet of Things (IoT), or High Performance Computing (HPC) these spatial and temporal differences also need to be accounted for in the logs, two papers addressed this [16, 32].

4.5 Log Analysis Goals

What the logs contain and their predictive outcome had a large correlation. Those logs containing general system information strove to predict overall system failure or faults [10, 34, 34, 37]. Event-based logs were focused on improving the efficiency of the event runtime by suggesting alternative event order. Logs containing event process data or process metadata were focused mainly on the improvement of scheduling for performance load balancing while error based logs focused on the prediction of said errors in a real-time environment [17, 19, 25, 29, 32]. Network or IoT logs focused on the prediction of process order in order to detect malicious activity or to improve fault detection in a microservice system [13, 20, 22, 35].

4.6 Prevailing Limitations and Challenges

Among the research papers analyzed, many common limitations and challenges facing the field of log prediction were mentioned. Among them were consistency [17], formatting [19, 22], complexity [13, 14, 29, 31], volume [22], scale [7, 13, 14], log quality [19], fault location [40], and time [7, 16, 40]. The most predominant issues are analyzed below.

4.6.1 Log Formatting and Quality. A common issue when trying to predict errors by logs is analyzing the logs because of their variable format. Five of the research papers analyzed cited log formatting as one of the predominant issues with log prediction [12, 19, 22, 28, 34]. On top of formatting, four studies also implied that general log quality was also a major roadblock for log prediction [10, 19, 23, 28]. The general issue with log formatting is that there are no standard formats which make the logs difficult to parse and analyze the messages stored inside the logs. These logs can also come from a variety of locations, each separate log having a different format. Log quality also can also affect the usefulness of logs. Log quality is generally defined by location and quantity. For example, if logs are not strategically placed within the code, failures may occur without leaving any logs to analyze. On the other hand, if there are too many log lines, then it becomes more difficult to find the actual problem. This can lead to the problem of inconsistency, which is discussed in the next section.

4.6.2 Log Consistency. Six of the research papers found mentioned error consistency as a major problem in error prediction from logs [12, 15–17, 34, 38]. Log inconsistency can occur in many different ways:

(1) A problem may never show itself the same way twice [17]
(2) Different programs use logs in different ways, making it difficult for an algorithm to analyze logs coming from a completely different program [34]
(3) A statically sized time window to find correlated events could not be large enough and miss events that should be correlated together, giving an incorrect view of the overall flow of the program [16]
(4) Error events can occur randomly, without a logical flow path [12]
(5) The amount of information available in logs can make it difficult to find the correct logical flow path of a program [15]
(6) The system behavior can change over time [38]

In general, the log prediction solutions are specific to the program they were created for; however, using machine learning, many of the papers were able to create methods to bypass this problem.

4.6.3 Scale, Volume and Complexity. The most common challenges mentioned among the research papers analyzed included scale, volume, and/or complexity. The growth in size and complexity of modern programs is the main reason error detection and prediction became a necessity. This is because the volume of logs increased with size and complexity. When the volume of logs is increased, the flow paths (or log clusters) created by general log prediction algorithms increase in size [9]. Therefore log prediction methods that worked on smaller systems may not be able to function using real-world data from large-scale systems [7, 29].

4.7 General Approaches to Log Prediction

The general approach to the problem of log prediction taken by the studies, RQ6, is of deepest importance when it comes to analyzing where research in the field sits. This can be broken down into three general categories: the method employed in the prediction (including any relevant algorithms), the validation of the method's effectiveness, and the scale of results they were able to achieve.

4.7.1 Methods and Algorithms. According to Kazmi in [20], most log prediction is based off of historical log data, using one of the following methodologies:

(1) Model-based
(2) Code-based
(3) Statistical Analysis-based
(4) Rule-based
(5) Markov Chain-based
(6) Artificial Neural Network-based
(7) Bayesian network-based

These classifications fit rather well with what the mapping study found, with a few important exceptions that will be discussed shortly. In general, however, there was no majority methodology found by the study. However, the methods used can be grouped into *"Niche Methodologies"* and *"Popular Methodologies"*, based off whether they were used as the primary method in at least 10% of the papers found by the study. Finally, there are methodologies that do not fit into a category, either because they are a hybrid of multiple categories or could not be classified; these will be discussed last.

Three of the options fall under *Niche Methodologies*. The first option, using *statistical analysis* to predict from log files, was found in only two papers [12, 20]. Next, basing the prediction off of *Markov chains* is also rather uncommon, with only two papers relying on different versions of the concept [31, 38]. Finally, using a *Bayesian network* for log prediction seems to have almost entirely disappeared, being used in only one paper [29].

The remaining four categories (as well as one additional category) fall under the *Popular Methodologies*: these are relying on a Model-based, Code-based, Rule-based, Artificial Neural Network, or a Support Vector Machine.

Three papers used a prediction schema that was based on some kind of *model* of the system under study to predict from log files [12–14]. Of these three, [13] and [14] were from the same author extending an initial study. Discounting these, no technology was used to create the model by more than one paper.

The next, *code-based*, requires some explanation before one will understand what the method is doing. A code-based predictive algorithm utilizes a codebook of identified error sequences [20]. This method is used by three papers [24, 32, 37]. Of these, [24] and [32] utilize a decision tree to create the codebook.

Rule-based prediction, in short, is utilizing a few guidelines that show what in a log indicates an error [20], meaning that the delineation between it and a code-based approach is occasionally somewhat hazy. This methodology is comparatively common, with four papers [16, 18, 23, 40]. While instinct may suggest that the rules used by this approach to detect errors must be written by a human, this is only the case in [23]; the other three employed various methodologies for the computer to extract the rules itself.

Applying an *Artificial Neural Network* to learn what an error looks like [20] was more common than the previous three methods, at four papers total [7, 8, 10, 35]. The preferred network was generally a Long Short-Term Memory (LSTM) neural network. LSTMs are Recurrent Neural Networks (RNN) augmented with memory and logic gates, known to retain long-term memory of short-term data chains that represent events correlated in time and space [7], capable of "forgetting" information which is irrelevant to a problem [35].

Finally, *Support Vector Machine (SVM)* being applied to predict from logs were surprisingly common, occurring in four of the found papers [9, 11, 15, 28]. Given that this is the case, what is an SVM? An SVM is a data mining algorithm that identifies functions to classify input data into categories, guaranteeing the maximum margin between different classes [36]. Thanks to this, it could be viewed as a subset of the rule-based approach (viewing the functions as rules), or possibly the code-based approach (considering the functions to be a codebook). However, thanks to having as many examples as the other popular approaches fully on their own, the use of an SVM has been split into its own category.

Aside from these two groups, certain papers could be classified into multiple or no categories. Two papers fell into multiple categories [19, 22]. The first paper combined statistical analysis and an Artificial Neural Network to predict anomalies in a cyber-physical power system [22], while the second manually selected either a multivariate linear regression model, multivariate polynomial model, a linear neural network, or a back-propagation neural network as the prediction model, based on which gave the best results [19].

Then, there are those papers that do not seem to fall under a category, with their reasons specified. In [39], the author compared multiple different methodologies, including a rule-based classifier, an SVM, and two versions of using the Nearest Neighbor method to create a set of correlation rules between fatal and nonfatal events [39]. For [34], the paper was itself a mapping study, leaving it hard to classify under just one category. Cuzzocrea et. all, in [6], state that they used clustering and time-series algorithms for their prediction but are unclear regarding exactly how these were worked and thus what category they fell under. Similarly, from reading [26], the methodology employed was difficult to deduce, though it seems to be a rule-based classifier. In contrast, Sahoo et. all compared a Bayesian network and rule-based methods, as well as a time-series model [30].

4.7.2 Validation of Methods. Validation of the study's results, in contrast with the above, is substantially more easily categorized. There are, in total, two different categorization methods: either by testing the proposed method head-to-head against other methods or by testing the method on a specified dataset. In addition, there is a small set of papers that do not adhere to these methodologies.

There are, in total, twelve papers which validated by comparing the results their method got against those garnered by other methods on the same data set [10, 12, 17, 22, 23, 26, 28, 31, 32, 35, 37, 39, 40]. In contrast, sixteen papers tested their proposed method on a specified data set [6–9, 11, 13–16, 18–20, 24, 29, 30, 38] to ensure that it achieved acceptable results.

This leaves two papers that do not follow these bulk categories. The first, [34], does not have any validation to perform since it is a mapping study, making it impossible to categorize under this schema. Then other, [28], instead compares its results to those garnered by other studies, making it similar to comparison to methodologies but still distinct from this.

4.7.3 Results of Methods. A few papers we analyzed provided rather strong comparisons of the different methods. In [22], the authors compared a number of approaches when validating that their Ensemble Prediction Algorithm Based on Time series (EPABT). When analyzed using mean absolute error, mean square error, and mean absolute percentage error, the results average to the following ordering, from best to worst: ensemble learning, gene expressive programming, artificial neural network, moving average, autoregressive, and finally source vector machine. Similarly, in [26], the authors ran extensive tests on 12 different algorithms for log prediction, determining that using a RandomTree (a randomized decision tree) and RandomForest (an ensemble classifier) methods were the most consistent across all metrics, with a Bayesian network and multiple decision tree implementations proving themselves rather poor on the whole in comparison. Finally, in [39], Zhang and Sivasubramaniam compared their customized nearest neighbor solution against a rule-based classifier called RIPPER and an SVM. These both proved to suffer intensely in performance as the "prediction window" (the time they are predicting into the future, of which the time they get to observe the system's past is a function) shrinks; this was not, however, an issue with their customized nearest neighbor method.

4.8 Future Directions

Most of the papers we investigated proposed future research directions in their conclusion sections. Upon analyzing those, we found three general categorizations for the future research direction on failure prediction. The first type of direction specified on those papers is the technical improvements of the prediction methods and investigating any discrepancies in the outcome. For instance, [18] mentioned an adaptive window size approach could be applied to reduce the training cost. Similarly, [32] pointed out an opportunity for further preprocessing of log files to achieve better results.

The second type is examining the approaches with different machine learning algorithms and extending them for better adjustment. In [34] authors are urging future researches to answer the question of whether or not LSTM can automatically derive relevant features from collections of trace prefixes to preclude sophisticated feature engineering. Replacement multilayer perceptron and radial basis function with Bayesian models can be explored further to achieve robustness to variation of data [28].

Finally, the third and most common type of future direction is the adaptation of existing works for more versatile environments, including logs from diverse sources. Although most of the research works have been done for a single environment addressing only one particular type of log, authors mentioned their urge to extend the works for multiple different contexts [17–20, 28]. Writers of [19] stated a possible extension of their work to include GPU jobs along with the existing prediction for parallel jobs. Predicting future alarms for telecommunication systems, as discussed in [20], can be adjusted for heterogeneous networks with a variety of vendors. Further experiments can be done on predicting failures in HPC systems using a variety of RAS (Reliability, Availability and Serviceability) logs from supercomputing centers [18].

Some other research directions do not fall into any of these three categories. However, they provide interesting insights into future trends. In [26], the authors mentioned the possibility of estimating the risk of failure instead of a binary failure/no failure prediction. The prediction performance for systems with limited data and yet unseen systems can also be evaluated [32].

4.9 Threats to Validity

The main threat to the validity of the mapping study is the exclusion of papers relevant to the topic in question: failure prediction using log analysis. We tried to mitigate this as much as possible by having our query be broad to include as many relevant papers as possible. However, it is still possible to miss some of the related papers in the search results due to versatile expressions and limitations of the search engine of underlying indexers. To tone down that issue, we further explored the related works of each paper filtered from the search results. The manual filtering process might be prone to wrong evaluation due to human errors, and that might cause omission of some relevant papers. This was addressed by reviewing the abstracts of each paper by multiple authors and considering the paper for a thorough analysis if any of the authors found it relevant.

5 CONCLUSION

Prediction based on log files, in many disciplines, can enable avoidance of costly software bugs, hardware failures, or network attacks before they impact a developer or business. Doing such a prediction automatically is a complicated endeavor, thanks to the diverse formats of the logs themselves and inconsistent information provided by them. This mapping study approached the discipline of log prediction as a whole, managing to acquire 30 papers on log prediction and classify the approaches taken into one of 8 different types. Of these classes, the most common were applying a model-based approach, code-based approach, rule-based approach, Artificial Neural Network, or a Support Vector Machine, with no real majority belonging to any category. In addition, it was discerned that research in this field, while high during the mid-2010s, is starting to drop, in spite of a number of avenues of study remaining open. One such future direction is adapting one of the many methodologies from the platform-specific versions proposed thus far to a platform-independent approach.

Our future research goal is to develop an automated error log resolution system by utilizing external knowledge sources like StackOverflow and Github. This survey helped us to identify existing work in the area of failure prediction. Besides, we have done similar mapping studies on log and code analysis [3, 4, 33] and constraint consistency checking [5] to achieve a better understanding of our future research goal.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 1854049 and a grant from Red Hat Research https://research.redhat.com.

REFERENCES

[1] Ahmed, Bestoun S and Bures, Miroslav and Frajtak, Karel and Cerny, Tomas. 2019. Aspects of quality in Internet of Things (IoT) solutions: A systematic mapping study. *IEEE Access* 7 (2019), 13758–13780.

[2] Bures, Miroslav and Cerny, Tomas and Ahmed, Bestoun S. 2018. Internet of things: Current challenges in the quality assurance and testing methods. In *International Conference on Information Science and Applications*. Springer, 625–634.

[3] Vincent Bushong, Russell Sanders, Jacob Curtis, Mark Du, Tomas Cerny, Karel Frajtak, Miroslav Bures, Pavel Tisnovsky, and Dongwan Shin. 2020. On Matching Log Analysis to Source Code: A Systematic Mapping Study. In *International Conference on Research in Adaptive and Convergent Systems(RACS '20) (RACS '20)*. ACM, New York, NY, USA, 1–6. https://doi.org/10.1145/3400286.3418262

[4] Tomas Cerny, Jan Svacina, Dipta Das, Vincent Bushong, Miroslav Bures, Pavel Tisnovsky, Karel Frajtak, Dongwan Shin, and Jun Huang. 2020. On Code Analysis Opportunities and Challenges for Enterprise Systems and Microservices. *IEEE Access* (2020), 1–22. https://doi.org/10.1109/ACCESS.2020.3019985

[5] Tomas Cerny, Andrew Walker, Vincent Bushong, Dipta Das, Karel Frajtak, Miroslav Bures, and Pavel Tisnovsky. 2020. Mapping Study on Constraint Consistency Checking in Distributed Enterprise Systems. In *International Conference on Research in Adaptive and Convergent Systems(RACS '20) (RACS '20)*. ACM, New York, NY, USA, 1–8. https://doi.org/10.1145/3400286.34182571

[6] Alfredo Cuzzocrea, Francesco Folino, Massimo Guarascio, and Luigi Pontieri. 2019. Predictive monitoring of temporally-aggregated performance indicators of business processes against low-level streaming events. *Information Systems* 81 (2019), 236–266.

[7] Anwesha Das, Frank Mueller, Charles Siegel, and Abhinav Vishnu. 2018. Desh: Deep Learning for System Health Prediction of Lead Times to Failure in HPC. In *Proceedings of the 27th International Symposium on High-Performance Parallel and Distributed Computing (HPDC '18)*. Association for Computing Machinery, New York, NY, USA, 40–51. https://doi.org/10.1145/3208040.3208051

[8] Željko Deljac, Mirko Randić, and Gordan Krčelić. 2016. A Multivariate Approach to Predicting Quantity of Failures in Broadband Networks Based on a Recurrent Neural Network. *Journal of Network and Systems Management* 24, 1 (01 Jan 2016), 189–221. https://doi.org/10.1007/s10922-015-9348-6

[9] R Wesley Featherstun and Errin W Fulp. 2010. Using Syslog Message Sequences for Predicting Disk Failures.. In *LISA*.

[10] F. Folino, G. Folino, M. Guarascio, and L. Pontieri. 2019. Learning Effective Neural Nets for Outcome Prediction from Partially Labelled Log Data. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*. 1396–1400.

[11] Ilenia Fronza, Alberto Sillitti, Giancarlo Succi, Mikko Terho, and Jelena Vlasenko. 2013. Failure prediction based on log files using random indexing and support vector machines. *Journal of Systems and Software* 86, 1 (2013), 2–11.

[12] Ilenia Fronza, Alberto Sillitti, Giancarlo Succi, and Jelena Vlasenko. 2011. Failure Prediction based on Log Files Using the Cox Proportional Hazard Model.. In *SEKE*. 456–461.

[13] Song Fu and Cheng-Zhong Xu. 2007. Exploring Event Correlation for Failure Prediction in Coalitions of Clusters. In *Proceedings of the 2007 ACM/IEEE Conference on Supercomputing (SC '07)*. Association for Computing Machinery, New York, NY, USA, Article 41, 12 pages. https://doi.org/10.1145/1362622.1362678

[14] Song Fu and Cheng-Zhong Xu. 2010. Quantifying event correlations for proactive failure management in networked computing systems. *Journal of parallel and distributed computing* 70, 11 (2010), 1100–1109.

[15] Errin W Fulp, Glenn A Fink, and Jereme N Haack. 2008. Predicting Computer System Failures Using Support Vector Machines. *WASL* 8 (2008), 5–5.

[16] Ana Gainaru, Franck Cappello, Joshi Fullop, Stefan Trausan-Matu, and William Kramer. 2011. 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 (SLAML '11)*. Association for Computing Machinery, New York, NY, USA, Article 4, 8 pages. https://doi.org/10.1145/2038633.2038637

[17] A. Gainaru, F. Cappello, M. Snir, and W. Kramer. 2012. Fault prediction under the microscope: A closer look into HPC systems. In *SC '12: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*. 1–11.

[18] J. Gu, Z. Zheng, Z. Lan, J. White, E. Hocks, and B. Park. 2008. Dynamic Meta-Learning for Failure Prediction in Large-Scale Systems: A Case Study. In *2008 37th International Conference on Parallel Processing*. 157–164.

[19] Z. Hou, S. Zhao, C. Yin, Y. Wang, J. Gu, and X. Zhou. 2019. Machine Learning Based Performance Analysis and Prediction of Jobs on a HPC Cluster. In *2019 20th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT)*. 247–252.

[20] A. S. Kazmi. 2011. Application of statistical sampling to predict faults from real time alarm data. In *2011 IEEE 14th International Multitopic Conference*. 290–295.

[21] Masanari Kondo, Cor-Paul Bezemer, Yasutaka Kamei, Ahmed E. Hassan, and Osamu Mizuno. 2019. The impact of feature reduction techniques on defect prediction models. *Empirical Software Engineering* 24, 4 (Aug. 2019), 1925–1963. https://doi.org/10.1007/s10664-018-9679-5

[22] Q. Li, S. Meng, S. Zhang, M. Wu, J. Zhang, M. Taleby Ahvanooey, and M. S. Aslam. 2019. Safety Risk Monitoring of Cyber-Physical Power Systems Based on Ensemble Learning Algorithm. *IEEE Access* 7 (2019), 24788–24805.

[23] T. . Y. Lin and D. P. Siewiorek. 1990. Error log analysis: statistical modeling and heuristic trend analysis. *IEEE Transactions on Reliability* 39, 4 (1990), 419–432.

[24] Fabrizio Maria Maggi, Chiara Di Francescomarino, Marlon Dumas, and Chiara Ghidini. 2014. Predictive Monitoring of Business Processes. In *Advanced Information Systems Engineering*, Matthias Jarke, John Mylopoulos, Christoph Quix, Colette Rolland, Yannis Manolopoulos, Haralambos Mouratidis, and Jennifer Horkoff (Eds.). Springer International Publishing, Cham, 457–472.

[25] Nijat Mehdiyev, Joerg Evermann, and Peter Fettke. 2020. A Novel Business Process Prediction Model Using a Deep Learning Method. *Business & Information Systems Engineering* 62, 2 (01 Apr 2020), 143–157. https://doi.org/10.1007/s12599-018-0551-3

[26] N. Nakka, A. Agrawal, and A. Choudhary. 2011. Predicting Node Failure in High Performance Computing Systems from Failure and Usage Logs. In *2011 IEEE International Symposium on Parallel and Distributed Processing Workshops and Phd Forum*. 1557–1566.

[27] Kai Petersen, Sairam Vakkalanka, and Ludwik Kuzniarz. 2015. Guidelines for conducting systematic mapping studies in software engineering: An update. *Information and Software Technology* 64 (Aug. 2015), 1–18. https://doi.org/10.1016/j.infsof.2015.03.007

[28] Barbara Russo, Giancarlo Succi, and Witold Pedrycz. 2015. Mining system logs to learn error predictors: a case study of a telemetry system. *Empirical Software Engineering* 20, 4 (Aug. 2015), 879–927. https://doi.org/10.1007/s10664-014-9303-2

[29] Hamid Saadatfar, Hamid Fadishei, and Hossein Deldari. 2012. Predicting Job Failures in AuverGrid Based on Workload Log Analysis. *New Generation Computing* 30, 1 (01 Jan 2012), 73–94. https://doi.org/10.1007/s00354-012-0105-z

[30] R. K. Sahoo, A. J. Oliner, I. Rish, M. Gupta, J. E. Moreira, S. Ma, R. Vilalta, and A. Sivasubramaniam. 2003. Critical Event Prediction for Proactive Management in Large-Scale Computer Clusters. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '03)*. Association for Computing Machinery, New York, NY, USA, 426–435. https://doi.org/10.1145/956750.956799

[31] F. Salfner, M. Schieschke, and M. Malek. 2006. Predicting failures of computer systems: a case study for a telecommunication system. In *Proceedings 20th IEEE International Parallel Distributed Processing Symposium*. 8 pp.–.

[32] Andreas Schörgenhumer, Mario Kahlhofer, Paul Grünbacher, and Hanspeter Mössenböck. 2019. Can We Predict Performance Events with Time Series Data from Monitoring Multiple Systems?. In *Companion of the 2019 ACM/SPEC International Conference on Performance Engineering (ICPE '19)*. Association for Computing Machinery, New York, NY, USA, 9–12. https://doi.org/10.1145/3302541.3313101

[33] Jan Svacina, Jackson Raffety, Connor Woodahl, Stone Brooklynn, Tomas Cerny, Miroslav Bures, Karel Frajtak, Dongwan Shin, and Pavel Tisnovsky. 2020. On Vulnerability and Security Log analysis: A Systematic Literature Review on Recent Trends. In *International Conference on Research in Adaptive and Convergent Systems(RACS '20) (RACS '20)*. ACM, New York, NY, USA, 1–6. https://doi.org/10.1145/3400286.3418261

[34] Irene Teinemaa, Marlon Dumas, Marcello La Rosa, and Fabrizio Maria Maggi. 2019. Outcome-Oriented Predictive Process Monitoring: Review and Benchmark. *ACM Trans. Knowl. Discov. Data* 13, 2, Article 17 (March 2019), 57 pages. https://doi.org/10.1145/3301300

[35] Pin Wu, Zhihui Lu, Quan Zhou, Zhidan Lei, Xiaoqiang Li, Meikang Qiu, and Patrick C.K. Hung. 2019. Bigdata logs analysis based on seq2seq networks for cognitive Internet of Things. *Future Generation Computer Systems* 90 (2019), 477 – 488. https://doi.org/10.1016/j.future.2018.08.021

[36] Xindong Wu, Vipin Kumar, J. Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J. McLachlan, Angus Ng, Bing Liu, Philip S. Yu, Zhi-Hua Zhou, Michael Steinbach, David J. Hand, and Dan Steinberg. 2008. Top 10 algorithms in data mining. *Knowledge and Information Systems* 14, 1 (01 Jan 2008), 1–37. https://doi.org/10.1007/s10115-007-0114-2

[37] I. Yagoub, M. A. Khan, and L. Jiyun. 2018. IT Equipment Monitoring and Analyzing System for Forecasting and Detecting Anomalies in Log Files Utilizing Machine Learning Techniques. In *2018 International Conference on Advances in Big Data, Computing and Data Communication Systems (icABCD)*. 1–6.

[38] Kenji Yamanishi and Yuko Maruyama. 2005. Dynamic Syslog Mining for Network Failure Monitoring. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining (KDD '05)*. Association for Computing Machinery, New York, NY, USA, 499–508. https://doi.org/10.1145/

1081870.1081927

[39] Y. Zhang and A. Sivasubramaniam. 2008. Failure prediction in IBM BlueGene/L event logs. In *2008 IEEE International Symposium on Parallel and Distributed Processing*. 1–5.

[40] Z. Zheng, Z. Lan, R. Gupta, S. Coghlan, and P. Beckman. 2010. A practical failure prediction with location and lead time for Blue Gene/P. In *2010 International Conference on Dependable Systems and Networks Workshops (DSN-W)*. 15–22.

[41] Z. Zheng, Z. Lan, B. H. Park, and A. Geist. 2009. System log pre-processing to improve failure prediction. In *2009 IEEE/IFIP International Conference on Dependable Systems Networks*. 572–577.