DOI: 10.2507/34th.daaam.proceedings.xxx

METHODS AND ALGORITHMS FOR ANALYSING AND PREDICTING ERRORS IN SERVER SYSTEMS

Plamena Koleva-Stoynova & Velko Iltchev

This Publication has to be referred as: Koleva-Stoynova, P[lamena] & Iltchev, V[elko] (2023). Methods and algorithms for analyzing and predicting errors in server systems, Proceedings of the 34th DAAAM International Symposium, pp.xxxx-xxxx, B. Katalinic (Ed.), Published by DAAAM International, ISBN 978-3-902734-xx-x, ISSN 1726-9679, Vienna, Austria
DOI: 10.2507/34th.daaam.proceedings.xxx

Abstract

Logging is important in monitoring and predicting server failures. The information is stored in log files, which contains (usually as plain text) information about messages, error reports, file requests and file transfers. Reasons for logging are diagnostic, audit, capturing stack traces or data. The purpose of this work is to develop a software architecture for analysis and prediction of errors in server systems. First, messages are collected from client servers and stored on the main server. Then a message parser recognizes the lexical and syntactic structure by using context-free grammar. Key components are stored in a relational database. A synthesizer is connected to a relational database. It is used to recreate the original messages when a user needs to read them. The most important part of this work is the recognition of interconnections between messages and environment with the help of artificial intelligence, which will thereafter be used to predict failures.

Keywords: logging; detection of errors; parsing; prediction.

1. Introduction

Logging is an essential practice, but it comes with potential issues that can interfere with its effectiveness. One common problem is the lack of storage space, leading to a low performance. Data acquisition generates a significant amount of information, consuming large storage space and impacting server performance. This information can also be sensitive, making proper handling crucial to avoid security issues. An illustrative instance is detailed in publication [1], wherein a swift expansion of the database can ultimately lead to disk capacity reaching full utilization, consequently giving rise to a severe issue.

Another issue is the log noise, where excessive or irrelevant logs make it challenging to diagnose actual problems effectively. For example, log entries might include messages that are not directly related to system health, security, or performance. These irrelevant messages obscure critical information. Additionally, excessive logging can impact an application's performance and response time. Furthermore, logs may not always provide sufficient context, making it harder to understand the full context of an issue. Finally, analysing a large volume of logs can be complex and time-consuming, which requires significant resources.

An approach to address these challenges involves the utilization of artificial intelligence techniques capable of discerning pertinent data from the unstructured log entries. An integral component of this solution is the anomaly detection. Through the analysis of individual data points, future errors can be readily anticipated and rectified promptly [2].

To perform an accurate analysis of the data log entries, log text parsing is necessary. It's noteworthy that in [3], a novel approach to anomaly detection based on log data known as NeuralLog is employed, which eliminates the necessity for log parsing.

The reason for creating this work is to develop an architecture for analysis and prediction of errors in server systems. The proposed paper is organized as follows: section 2 describes related to the research works, section 3 describes the proposed method with topology and the connection with neural network. The conclusion closes the article.

2. Related Works

2.1. The platform LogPAI

Using advanced algorithms for predicting and parsing methods, scientists from the Chinese University of Hong Kong developed a platform for collecting, parsing, and analysing logs from server systems. This project consists of large collection of system log datasets for AI-powered log analytics (loghub), a toolkit for automated log parsing (log parser), a log analysis toolkit for automated anomaly detection (loglizer) and log-based impactful problem identification using machine learning(log3C). All these are components of the same platform.

At first, logs are generated at runtime and aggregated into a centralized place with a data streaming pipeline, such as Flume and Kafka [4]. Then a log parser converts unstructured log messages into a map of structured events, based on which sophisticated machine learning models can be applied. [5] Afterwards the structured logs can be sliced into short log sequences through interval window, sliding window, or session window. Then, feature extraction is performed to vectorize each log sequence, for example, using an event counting vector. [6] Finally, the anomaly detection models are trained to check whether a given feature vector is an anomaly or not. [7]

2.2. Software Logstash

Another software example is Logstash. It is an open-source data collection engine with real-time pipelining capabilities. It can dynamically unify data from disparate sources and normalize the data into destinations. It serves as a crucial component in the ELK (Elasticsearch, Logstash, Kibana) stack, which is widely used for log management and data analysis. As a data collector, Logstash can ingest data from diverse sources such as files, syslog, redis, and beats. Once the data is collected, Logstash processes it through a series of filters in a pipeline. These filters can parse, structure, modify, and enrich the incoming data, making it more organized and suitable for analysis.

At first, inputs are used to get data into Logstash. These inputs are file, syslog, redis, beats. Then, filters process information in pipeline. Used filters are: grok, mutate, drop, clone and geoip. Grok-filter parses and structures the unstructured log. Mutate-filter performs general transformations on event fields, such as rename, remove or modify. Drop-filter drops an event completely. Clone-filter makes a copy of an event. Geoip-filter adds information about geographical location of IP addresses.

Outputs are the final phase of the Logstash pipeline. An event can pass through multiple outputs, but once all output processing is complete, the event has finished its execution.

With its flexible and configurable nature, Logstash allows users to handle a wide variety of data formats and sources, making it a powerful tool for managing large-scale log data and other data streams in real-time. [8]

2.3. Software Fluentd

Fluentd is another software example. It is an open-source data collector, which unifies the data collection and consumption for a better use and understanding of data. It collects, processes and aggregates logs. Fluentd was designed to work as a unified layer for logging a central component that aggregates data from multiple sources that vary formatted data in JSON objects and to different output destinations can forward. One of Fluentd's key strengths lies in its pluggable architecture, which allows users to extend its capabilities with custom plugins for data enrichment, filtering, and transformation. This enables seamless integration with different data storage and analytics systems, such as Elasticsearch. [9]

3. Proposed Method

The objective of this development is to define an architecture for the analysis and prediction of errors within server systems. The proposed approach delineates the interrelationship connecting client servers, the primary server, and a relational database. Client Servers establish a connection with the main server, facilitating the transmission of log messages from client servers to the primary server for aggregation. Utilizing a parser, log messages' lexical and syntactic structure is discerned through the application of context-free grammar. Consequently, facilitating the distribution and storage of key components into designated tables within the relational database is important. A synthesizer possesses authorization to access this database. In instances where a user seeks to access a specific message, the synthesizer reconstructs it by applying a template alongside pertinent components sourced from the database. Subsequently, using an artificial intelligence, the neural network identifies linkages between messages and their contextual surroundings, thereby enhancing comprehension of their origin and implications.

In the fourth figure presented below, the depicted approach can be observed.

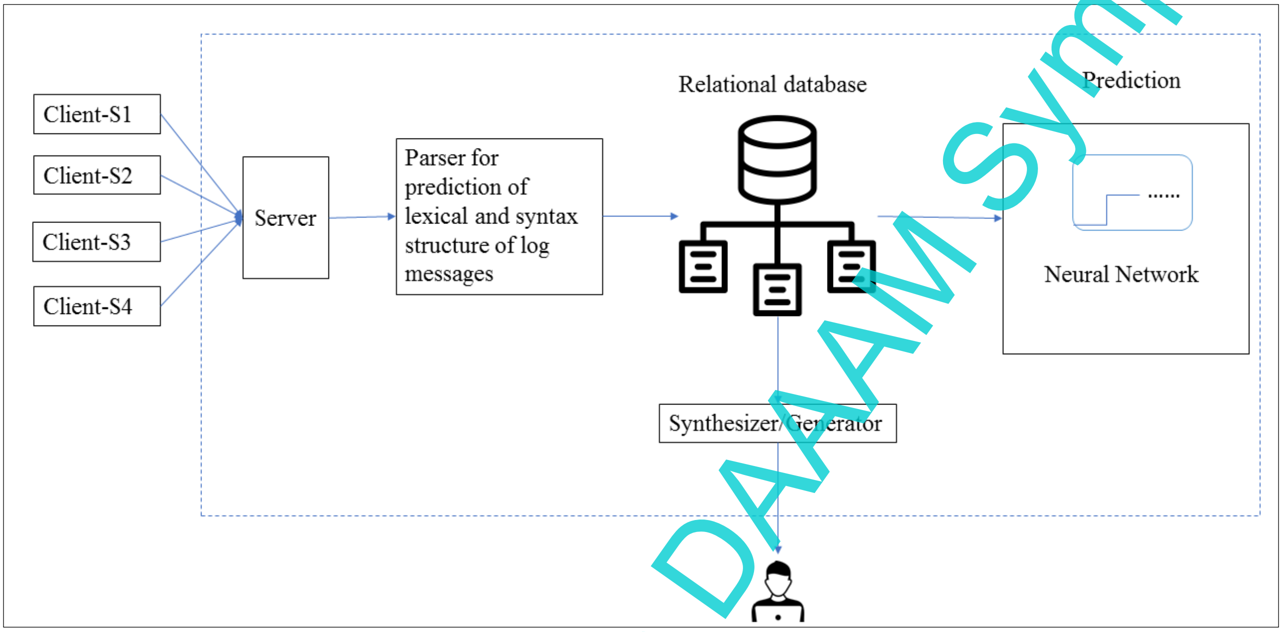

Fig. 1. Proposed Architecture

An essential aspect of this study involves investigating the interrelationships within this architecture and comprehending the behavioural patterns of data. The interplay between neural network analysis and a relational database offers a holistic approach to data-driven decision-making, prediction, and understanding complex systems. This integration capitalizes on the strengths of both methodologies, leading to more accurate, contextually aware analyses and predictions.

3.1. Extraction of facts

Important point in this work is the extraction of facts from messages. The process involves identifying and isolating specific structured information within the log data. This procedure encompasses the recognition of typical patterns within log message formats, achieved through techniques like regular expressions or advanced sequence modelling. Pertinent details such as timestamps, error codes, user identifiers, and IP addresses within log messages are targeted for extraction through algorithmic means. Log messages encompass a combination of structured and unstructured data. While structured data, such as timestamps and numerical values, is relatively straightforward to extract, unstructured data like error descriptions necessitates the application of advanced natural language processing methods. Fact extraction can leverage grammar expressions for recurring patterns, while custom rules can be tailored to capture domain-specific information by identifying distinct keywords or phrases. Extracted facts find utility across various analytical pursuits, including trend analysis, anomaly detection, predictive modelling, and system monitoring.

```
[appbf]/var/lxc/cXXXX/var/perp/app/bruteforce/mon/current

2021-06-08 20:04:06.932583 gotdata: {"ua":"Mozilla/5.0 (Windows NT 10.0;
Win64; x64) AppleWebKit/537.36(KHTML,like Gecko) Chrome/83.0.4087.0
Safari/537.36","app":"wordpress","ip":"1.2.3.4","type":"comments","path":"/
home/customer/www/autoexample.com/public_html","login":"unknown","host":"ww
w.autoexample.com"}
```

Fig. 2. Example of Log Message

The provided example on figure 2 serves as a representation of a simple textual format messages. Line 1 shows the name of file and its full path destination. Line 3 shows the plain text message with key components. To extract the fundamental elements from the message, using of the context-free grammar is needed. Therefore, the example will be depicted as following graph:

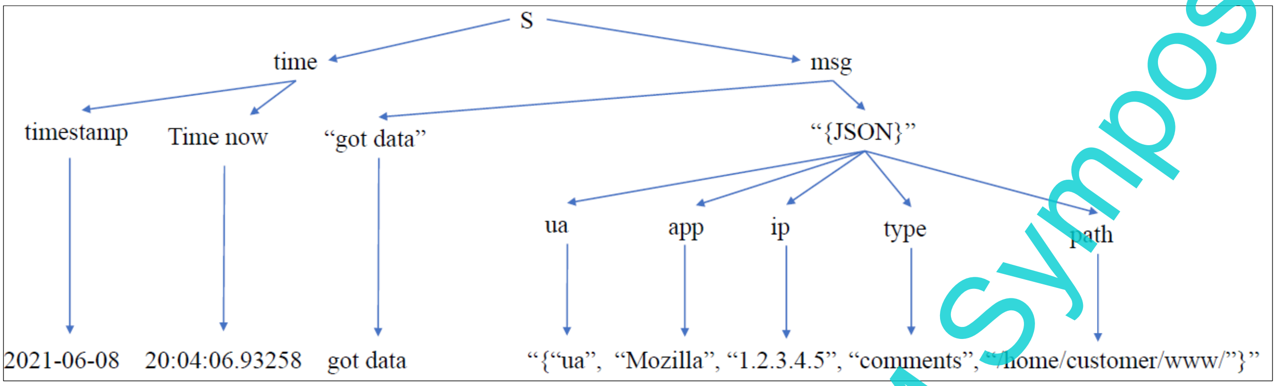

Fig. 3. Syntax tree of log message example

The represented syntax tree illustrates the message dissected into its constituent elements. The entire message is denoted by the abbreviation "S" and subsequently divided into two segments, namely "time" and "msg." The acronym "msg" signifies "message." Subsequently, the "time" segment is further subdivided into three components: "timestamp," "current time," and "data received" indicated by "got data." The data received is structured in JSON (JavaScript Object Notation) format, enhancing accessibility to key elements. For instance, for the "app" key within messages of the same type, the corresponding value imparts insights about the utilized application.

3.2. Allocation of facts

Another significant aspect within this study pertains to the dissemination of factual elements across a database. Numerous tables will be established for the purpose of retaining values derived from the log message. These tables will be linked via a primary identifier. This connectivity is advantageous for facilitating convenient retrieval during message reconstruction or data analysis endeavours. Thereafter, a neural network will be connected to this database with the aim to be trained to analyse and predict future errors.

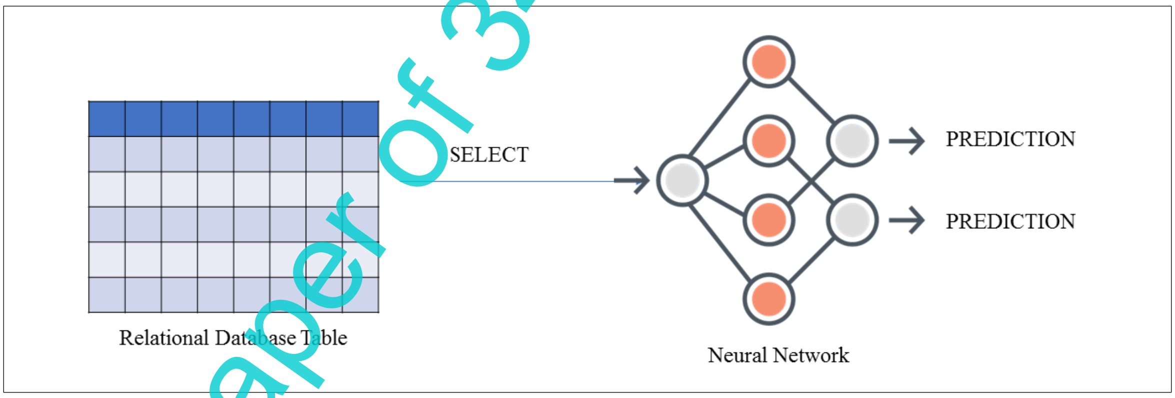

Fig. 4. Communication between table from database and neural network

The depiction in Figure 4 showcases the process. By utilizing the "Select" command, structured data within the database is extracted and enters the initial phase of the neural network. The neural network is pre-trained to assess outcomes generated by such commands. The message is then stored in an intermediate format, but with a distinct word order.

3.3. Prediction of future events

Collection of facts is needed for analytics of the whole picture. The use of time series forecasting with log messages involves the use of statistical and mathematical techniques to predict future patterns and trends within the log data. By treating log messages as sequential data points ordered by time, time series forecasting methods can be applied to anticipate potential future events, anomalies, or changes in the log data.

This approach leverages historical log message data to establish patterns and relationships over time. The collected information is then used to develop predictive models that can project how the log messages might evolve in the future. This could include predicting potential system failures, performance issues, or unusual activities based on past patterns and trends observed in the log messages.

Time series forecasting techniques, such as ARIMA (Autoregressive Integrated Moving Average), Exponential Smoothing, or more advanced machine learning algorithms like LSTM (Long Short-Term Memory) neural networks, can be applied to log messages. These methods analyse temporal dependencies, seasonal variations, and other recurring patterns to generate predictions.

The incorporation of time series forecasting with log messages is particularly valuable in proactive system maintenance, resource optimization, and anomaly detection. It aids administrators in identifying potential issues before they escalate, thus enhancing overall system reliability and performance. The method described herein finds application in the subsequent scholarly work titled “The importance of time series data filtering for predicting the direction of stock market movement using neural networks” [10].

Searching for errors in such huge files is slow and inefficient. The solution we offer, we can quickly find errors of one type that occurred in a certain time interval from certain servers. A neural network training will allow to analyse these errors and therefore predict future ones.

4. Conclusion and future work

The core issue discussed in this article pertains to server logging, specifically the challenge of excessive disk space use due to logging activities. This problem can be attributed to overly detailed logging, necessitating a focus on capturing essential information. Furthermore, the practice of log rotation, which entails the periodic creation of new log files accompanied by the compression or removal of older log files, is identified as another factor contributing to increased disk space consumption. Additionally, misconfigurations of logging levels, such as the use of INFO or DEBUG, may exacerbate this issue.

While logging remains a vital component of system monitoring, its storage in raw text format can impose a substantial space burden. To mitigate this, it is imperative to selectively extract and retain only the crucial data, subsequently storing it within a database. This structured approach enables the generation of templates, facilitating enhanced comprehension of log messages by clients seeking to discern the underlying issues.

This paper introduces a novel approach to analyse and forecast errors within server systems. To achieve this objective, an architecture is built to establish connections between client servers and the central server. A parser is employed to extract and disseminate information into a database, which serves as the foundation for making predictions regarding future errors. The primary emphasis of this research lies in the examination of interconnections within this architectural framework.

Future research will encompass the development of advanced machine learning models for more accurate error anticipation and classification. It will also focus on creating real-time predictive systems to minimize service disruptions. Interdisciplinary collaboration, involving fields like cybersecurity and data analytics, will yield innovative methods for improving system reliability. Research will explore human-machine interaction for enhanced error diagnosis and resolution, particularly in complex server environments. Additionally, scalability in cloud-based systems, privacy and security considerations, economic impact assessments, and open-source collaboration will be central to shaping the future of error analysis and prediction.

The exploration of methods and algorithms for analysing and predicting errors in server systems yields valuable insights into enhancing system reliability and performance. By meticulously investigating various techniques, this study contributes to the advancement of error detection, diagnosis, and prevention strategies. The integration of sophisticated algorithms and predictive models equips server administrators with proactive tools to mitigate disruptions, minimize downtime, and optimize system operations. As technology continues to evolve, the continuous refinement of these methods promises a more resilient and efficient server ecosystem, enabling organizations to uphold robust performance standards and deliver seamless user experiences.

5. Acknowledgments

This paper is dedicated to my grandmother, Ing. MSc. Paraskeva Ilieva, for her constant faith in my abilities. Her influence on my academic and personal growth has been profound, and I am grateful for the wisdom and support she has provided. This work stands as a tribute to her memory, which continues to shape and guide my endeavors.

6. References

[1] Korbar D, “Problem with rapid transaction log growth in SQL server databases,” in DAAAM International Vienna, 2010

[2] Min Du, Feifei Li, Guineng Zheng, Vivek Srikumar, “DeepLog: Anomaly Detection and Diagnosis from System Logs,” in CCS '17: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. New York, 2017

[3] Van-Hoang Le, Hongyu Zhang, “Log-based anomaly detection without log parsing,” in Proceedings of the 36th IEEE/ACM International Conference on Automated Software Engineering, Melbourne, 2021

[4] WangJun, Wang Wenhao, Chen Renfei, „Distributed Data Streams Processing Based on Flume/Kafka/Spark“ in 3rd International Conference on Mechatronics and Industrial Informatics (ICMII 2015), October 2015, Zhuhai, China

[5] A. Zbiciak and T. Markiewicz, "A new extraordinary means of appeal in the Polish criminal procedure: the basic principles of a fair trial and a complaint against a cassatory judgment," *Access to Justice in Eastern Europe*, vol. 6, p. 1–18, March 2023

[6] S. He, J. Zhu, P. He and M. R. Lyu, “Experience Report: System Log Analysis for Anomaly Detection,” in *27th IEEE International Symposium on Software Reliability Engineering, ISSRE 2016, Ottawa, ON, Canada, October 23-27, 2016*, 2016

[7] Shilin He, Qingwei Lin, Jian-Guang Lou, Hongyu Zhang, Michael R. Lyu, Dongmei Zhang, “Identifying impactful service system problems via log analysis,” in *ESEC/FSE 2018: Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 26 October 2018

[8] “Parsing Logs with Logstash,” [Online]. Available: https://www.elastic.co/guide/en/logstash/current/advanced-pipeline.html. Accessed on: 24 August 2023

[9] “What is Fluentd?,” Cloud Native Computing Foundation (CNCF), [Online]. Available: https://www.fluentd.org/architecture. Accessed on: 24 August 2023

[10] Ive Botunac, Ante Panjkota & Maja Matetic, “The importance of time series data filtering for predicting the direction of stock market movement using neural networks,” in *30th DAAAM international symposium on intelligent manufactoring and automation*, Vienna, Austria, 2019