IJRAR.ORG E-ISSN: 2348-1269, P-ISSN: 2349-5138

INTERNATIONAL JOURNAL OF RESEARCH AND ANALYTICAL REVIEWS (IJRAR) | IJRAR.ORG

An International Open Access, Peer-reviewed, Refereed Journal

AI POWERED SERVER LOG MANAGEMENT SOFTWARE

"AI-Powered Server Log Expertise "

[1]Shaik Salma Begum, [2] Amirisetty Deepak Venkat, [3] S Pranay Kiran, [4] Akshay Jayesh, [5]Kesava M

[1]Assistant Professor, Dept. of Computer Science and Engineering, Presidency University, Karnataka, India,
[2345]UG Student, Dept. of Computer Science and Engineering(Data Science), Presidency University, Karnataka, India
[1]Computer Science Engineering (Data Science)
[1]Presidency University, Bangalore, India

Abstract :

In software engineering, logging traditionally aids in root cause analysis and failure identification. Recent advancements in automated log file analysis have expanded its utility to include real-time system health monitoring, user behavior analysis, and domain knowledge extraction. However, the lack of a standardized format remains a significant obstacle. Focusing on the last five years, this paper summarizes key research topics, categorizes log files used in studies, and offers an informative foundation for future investigators, providing insights into diverse log data applications.

Keywords: Artificial intelligence, Server log management, Error detection, Log analysis, Automated analysis, Log file standardization, Django framework, User behavior.

I. INTRODUCTION

Systems with a lot of software generate logs for debugging. On the basis of log data, numerous deep learning models have recently been developed to automatically detect system anomalies. Usually, these models make very high detection accuracy claims. For instance, on the widely used dataset, the majority of models show an F-measure greater than. In this research, we do an extensive examination of five state-of-the-art deep learning-based models for detecting system anomalies using four public log datasets in order to gain a comprehensive understanding of how distant we are from solving the challenge of log-based anomaly detection. Our studies concentrate on a number of model evaluation factors, such as the choice of training data, data grouping, class distribution, data noise, and early detection capability. Our findings indicate that each of these factors significantly affects the evaluation and that not all of the models under study perform as expected. Log-based detection remains an unsolved problem. We also make some suggestions for potential future work based on our findings.

II. RELATED WORK

He et al. (2018) presented an improved KNN-based efficient log anomaly detection method using automatically labeled samples.

This study suggests a way to deal with issues such as high-dimensional data, unlabeled logs, and data imbalance that arise when using KNN for log anomaly detection. It presents methods for effective neighbor searching, automatic labeling, and dimensionality reduction.

Wang et al. (2020) present a log-based anomaly detection method with automatic K neighbor selection and efficient neighbor searching.

Through the application of min-hash and MVP-tree algorithms for neighbor searching and the investigation of automatic k neighbor selection to accommodate varying data distributions, this work aims to improve the efficiency of KNN-based log anomaly detection.

III. PROPOSED METHOD

In this project we will train different software errors and possible solutions with machine or deep learning (Artificial Intelligence) algorithms and then generate a trained model.

Steps in Implementation:

Data Collection:
Gather a variety of log files from different server sources to guarantee accurate timestamps and a wide coverage of scenarios.

Preparing Data:
Clear and standardize log data, taking into account variations in format and content. Extrapolate pertinent data, such as timestamps, error messages, and application IDs.

Labeling Data through Supervised Learning:
Mark a portion of the log entries by hand as normal or anomalous. Utilizing labeled data, train the K-Nearest Neighbors (KNN) algorithm.

KNN Model Instruction:
Use Django to implement a supervised machine learning model. Using labeled log data, train the KNN algorithm to identify common behavior patterns.

Finding anomalies:
To find deviations from learned patterns, run the entire log dataset through the trained KNN model. Indicate which entries could be mistakes.

Error Recording and Creating Dashboards:
When anomalies are found, automatically create error logs including timestamps, error kinds, and details about the affected application. Create dynamic Django dashboards that summarize statistics and error trends.

Integration of Real-time Monitoring:
Use real-time log monitoring to find anomalies quickly, record them, and send out alerts or notifications.

Development of User Interfaces:
Use Django to create an intuitive user interface for displaying log data and error analytics. Provide search, filtering, and focusing tools for particular error details.

Deployment and Scalability:

Install the AI-driven log management system on the server, making sure it can grow to accommodate substantial volumes of log data.

Upkeep and Record-Keeping:

Keep thorough records of the system architecture, data flow, and maintenance guidelines. Update the system often to take into account modifications to log patterns or server configurations.

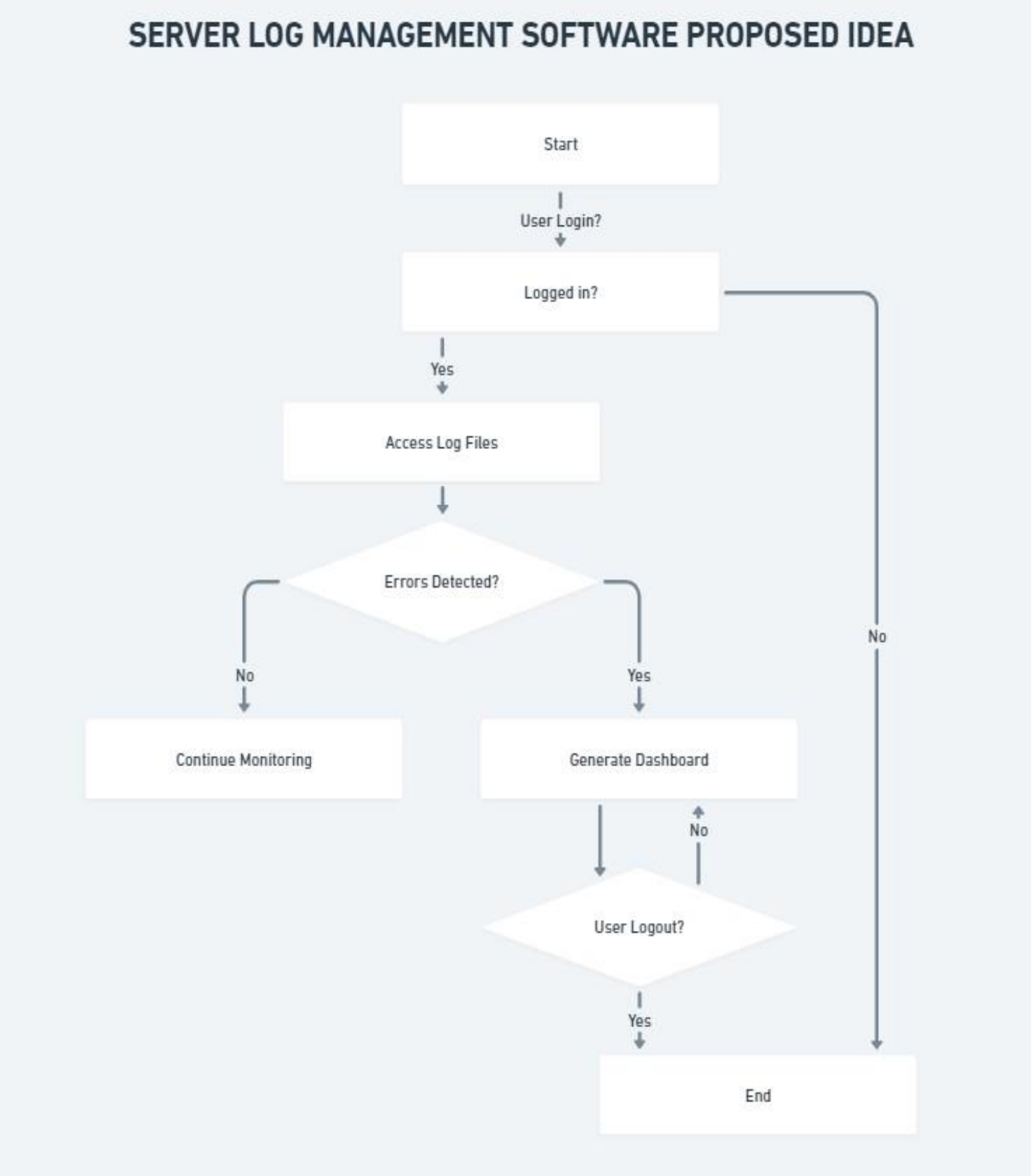

Fig. 1 server log management software flow

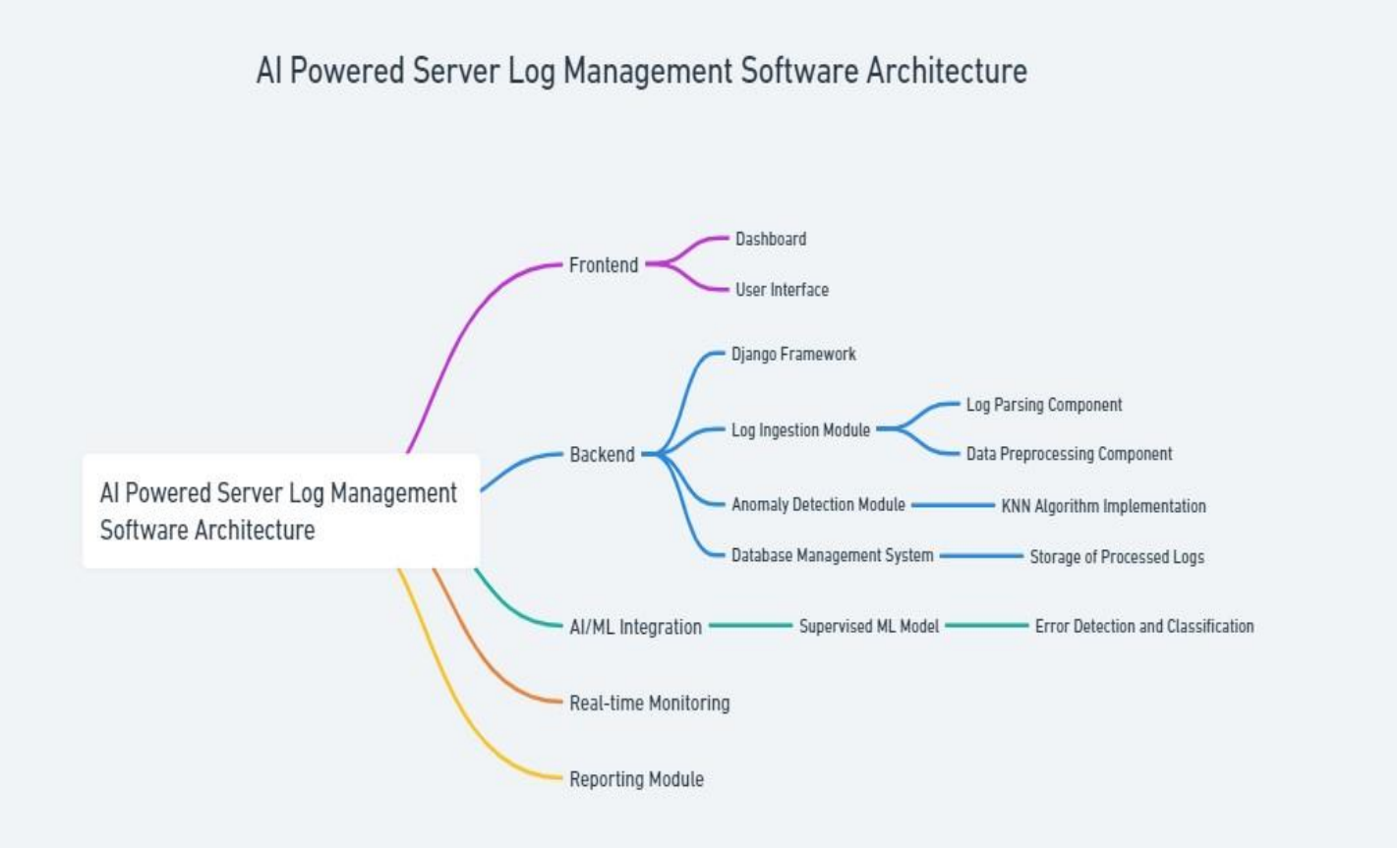

Fig.2 Architectural Design

ACKNOWLEDGMENT

Jon Dixon (1996), Peter N"uchter (1996), Juergen von Hagen (2000), and Michael Shell (2001-2002) for their invaluable efforts in developing and maintaining the IEEE LaTeX style files. The meticulous work on the LaTeX style files has greatly facilitated the preparation of this project template. The contributions of these individuals are acknowledged with gratitude.

IV. CONCLUSION

In conclusion, our AI-powered server log management software, which uses the K-Nearest Neighbors (KNN) algorithm and is based on the Django framework, provides a reliable means of automating error detection and log analysis. Through methodical resolution of issues related to heterogeneous log formats and content, our system effectively detects anomalies in real-time, improving both system health monitoring and comprehensive understanding of user behavior. For thorough error analytics, the dynamic dashboards created from error logs offer an easy-to-use interface.

I. REFERENCES

[1] D. Yuan, S. Park, and Y. Zhou, ‘‘Characterizing logging practices in open source software,’’ in Proc. 34th Int. Conf. Softw. Eng. (ICSE), Jun. 2012,

[2] M. Fuller, E. Brighton, M. Schiewe, D. Das, T. Cerny, and P. Tisnovsky, ‘‘Automated error log resolution: A case study,’’ in Proc. 36th Annu. ACM Symp. Appl. Comput., Mar. 2021,

[3] X. Wang, D. Wang, Y. Zhang, L. Jin, and M. Song, ‘‘Unsupervised learning for log data analysis based on behavior and attribute features,’’ in Proc. Int. Conf. Artif. Intell. Comput. Sci., Jul. 2019,

[4] R. C. Raga and J. D. Raga, ‘‘A comparison of college faculty and student class activity in an online earning environment using course log data,’’ in Proc. IEEE SmartWorld Ubiquitous Intell. Comput. Adv. Trust. omput. Scalable, Dec. 2018,

[5] G. Qi, W. T. Tsai, W. Li, Z. Zhu, and Y. Luo, ‘‘A cloud-based triage log analysis and recovery amework,’’ Simul. Model. Pract. Theory, vol. 77, Aug. 2020.

[6] X. Tian, H. Li, and F. Liu, ‘‘Web service reliability test method based on log analysis,’’ in Proc. IEEE Int. Conf. Softw., Rel. Secur. Companion (QRS-C), Jul. 2017,

[7] D. Zou, H. Qin, and H. Jin, ‘‘UiLog: Improving log-based fault diagnosis by log analysis,’’ J. Comput. Sci. Technol., vol. 31, no. 5, 2016, doi: 10.1007/s11390-016-1678-7.

[8] S. Locke, H. Li, T. H. P. Chen, W. Shang, and W. Liu, ‘‘LogAssist: Assisting log analysis through log summarization,’’ IEEE Trans. Softw. Eng., early access, May 26, 2021, doi: 10.1109/TSE.2021.3083715.

[9] J. Cándido, M. Aniche, and A. Van Deursen, ‘‘Log-based software monitoring: A systematic mapping study,’’ PeerJ Comput. Sci., vol. 7, Oct. 2021

[10] D. Obrâbski and J. Sosnowski, ‘‘Log based analysis of software application operation,’’ Adv. Intell. Syst. Comput., vol. 987, Oct. 2020

[11] Q. Cao, Y. Qiao, and Z. Lyu, ‘‘Machine learning to detect anomalies in web log analysis,’’ in Proc. 3rd IEEE Int. Conf. Comput. Commun. (ICCC), Dec. 2017,

[12] B. Debnath, M. Solaimani, M. A. G. Gulzar, N. Arora, C. Lumezanu, J. Xu, B. Zong, H. Zhang, G. Jiang, and L. Khan, ‘‘LogLens: A real-time log analysis system,’’ in Proc. IEEE 38th Int. Conf. Distrib. Comput. Syst.(ICDS),July 2018

Similarity Index / Plagiarism Check

AI POWERED SERVER LOG MANAGEMENT SOFTWARE

ORIGINALITY REPORT

9%	7%	7%	2%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	export.arxiv.org Internet Source	4%
2	Anandakumar Haldorai, Suriya Murugan, Arulmurugan Ramu. "Evolution, challenges, and application of intelligent ICT education: An overview", Computer Applications in Engineering Education, 2020 Publication	2%
3	github.com Internet Source	2%
4	Bingming Wang, Shi Ying, Zhe Yang. " A Log-Based Anomaly Detection Method with Efficient Neighbor Searching and Automatic Neighbor Selection ", Scientific Programming, 2020 Publication	1%

Exclude quotes	On	Exclude matches	< 1 words
Exclude bibliography	On		