Internship Report : Log-Based Anomaly Detection using Machine Learning

Department of Computer Science, Sapienza University of Rome

July 7 – July 17, 2025

Mohammed Bekkouche
LabRI-SBA Lab., École Supérieure en Informatique de Sidi Bel Abbès (Algeria)

1. Introduction

This report summarizes the activities and outcomes of my internship at the Department of Computer Science, Sapienza University of Rome, Italy, from **July 7 to July 17, 2025**.

The internship focused on **Log-Based Anomaly Detection using Machine Learning**. It was funded by my institution, the **École Supérieure en Informatique de Sidi Bel Abbès (ESI-SBA)**, Algeria, to enhance research capacity and foster international cooperation.

2. Context and Motivation

Modern systems are increasingly evolving into large-scale infrastructures by scaling out to distributed architectures composed of thousands of commodity machines.

These distributed systems generate massive volumes of log data that are critical for system monitoring and maintenance. Logs capture system events and internal states during runtime, providing valuable insights into system behavior.

Developers can manually inspect logs to detect anomalies, the scale and complexity of modern systems make this task extremely challenging or even infeasible.

As a result, automated log analysis methods have become essential.

3. Objectives

- Deepen my expertise in applying machine learning techniques to system log analysis for anomaly detection.
- Explore recent tools and datasets relevant to the domain.
- Discuss potential research cooperations and future joint publications.
- Prepare to integrate these approaches into teaching and student projects at ESI-SBA.

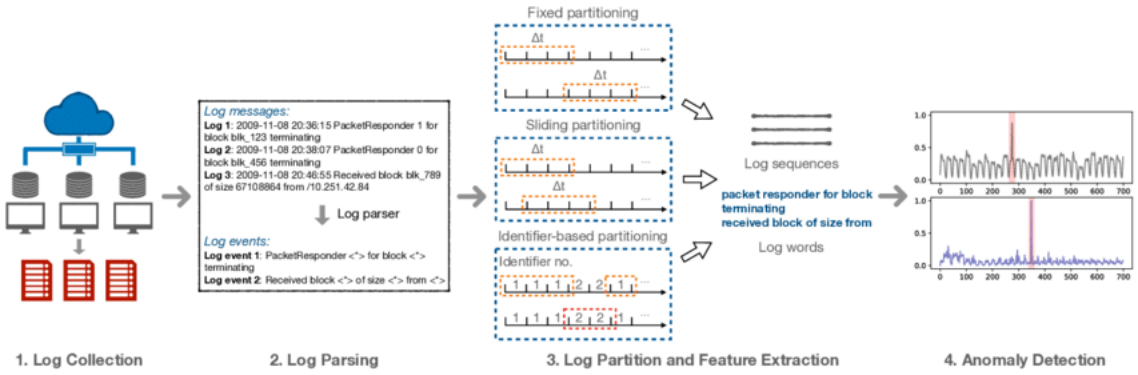

Figure 1: The Approach Used for Anomaly Detection [5, 3].

4. Log-Based Anomaly Detection

An anomaly is a behavior that goes against what the system is expected to do. It may indicate problems such as faults or unexpected conditions. Logs are sequences of messages recorded by the system in time order. They reflect events and internal states during operation, helping us understand how the system behaves.

Log-based anomaly detection aims to identify abnormal system behaviors, which can be early indicators of failures, by analyzing the log data generated during execution. It has become an important technique for ensuring system reliability and service quality.

5. Typical Challenges

Log analysis comes with improtant challenges:

- The formats are diverse, making them harder to handle.
- Logs include noise and repeated information.
- And manual inspection is time-consuming and prone to errors.

6. Machine Learning Approaches

In anomaly detection, there are two main learning approaches.

Supervised learning [2, 8, 1] relies on labeled data, where each instance is marked as normal or anomalous. This helps the model learn to distinguish between the two.

Unsupervised learning [11, 6, 9, 4, 7, 3], on the other hand, works without labels. It learns normal patterns from the data itself and is suitable when labeled data isn't available, which is common in real-world systems.

7. Detection Approach

The anomaly detection framework has four principal steps (see fig. 1): log parsing, feature extraction, model training, and anomaly detection. In the first step, log parsing, we transform unstructured log messages into structured formats by extracting event templates and separating

variable parts. For instance, a message like 'Received block X of size Y from Z' becomes a defined event pattern. Then comes feature extraction. We group logs into sequences using techniques like sliding or session windows. For each sequence, we create a feature vector that records how many times each event appears, forming a matrix of counts. This feature matrix is used to train a machine learning model that learns the normal behavior. In the final step, the trained model analyzes new log sequences and decides whether they represent normal activity or anomalies.

8. Common Models

Anomaly detection is applied to the feature vectors created earlier from log data. Most machine learning models give one prediction per log sequence, helping us spot which sequences are abnormal. A variety of models can be used — from classic ones like SVM, Random Forest, and Decision Tree, to clustering methods like K-Means and DBSCAN, and more advanced techniques like Autoencoders, LSTM, and Transformers. The choice depends on the type of data and whether labels are available.

Example: Autoencoder An example of a typical model used in unsupervised anomaly detection is the autoencoder. It's trained using only normal log sequences and learns to reconstruct them. If a new sequence is also normal, the reconstruction error stays low. But if the sequence is unusual, the model struggles to rebuild it, leading to a high error — which signals an anomaly. This makes autoencoders a powerful tool for detecting unexpected behavior in log data without needing labeled anomalies.

9. Case Study

This case study uses HDFS (Hadoop Distributed File System) [11, 12] log data from the Amazon EC2 platform.

The dataset contains over 11 million log messages. Each message contains a block ID, which allows us to group logs into sequences using session windows.

Feature vectors are then extracted from these sequences, producing 575,061 event count vectors. Among them, 16,838 are labeled as anomalies, providing a strong basis for testing anomaly detection models.

10. Example Results (LogAnomaly)

In this section, we present the results for an important machine-learning-based approach for anomaly detection, namely LogAnomaly [10], applied on the HDFS dataset. The results are reported in terms of Precision,Recall,and F1-Score, which are popular metrics used to evaluate anomaly detection models.

Precision represents the percentage of true anomalies among all predicted anomalies. Recall represents the percentage of actual anomalies that are correctly identified. The F1-Score is a balanced average between Precision and Recall.

The formulas for computing these metrics are :

$$\text{Precision} = \frac{TP}{TP + FP} \tag{1}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2}$$

$$\text{F1-Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3}$$

Where: **TP (True Positive)**: Correctly detected anomalies, **TN (True Negative)**: Correctly detected normal logs, **FP (False Positive)**: Normal logs incorrectly identified as anomalies, **FN (False Negative)**: Missed actual anomalies

The Precision, Recall, and F1-Score achieved by the LogAnomaly approach are 0.96, 0.94, and 0.95, respectively. This demonstrates the impressive accuracy attained by this approach.

3. Activities Carried Out

- Met with **Prof. Enrico Tronci** and his team to discuss research on anomaly detection and machine learning.
- Reviewed state-of-the-art techniques for preprocessing heterogeneous log data.
- Implemented and evaluated machine learning models, such as Isolation Forests and LSTM autoencoders, on example log datasets.
- Discussed combining statistical and rule-based methods for improved detection.
- Prepared outlines for a potential joint publication.

4. Outcomes

- Gained practical experience with recent tools and libraries (Python's Scikit-learn, Keras, and log parsing frameworks).
- Identified challenges such as noise and redundancy in log data, and studied solutions.
- Strengthened academic ties with the team at Sapienza University, opening avenues for future cooperations and student internships.

5. Conclusion

To sum up, logs are an important resource for understanding system behavior and detecting issues. Machine learning brings a significant boost to anomaly detection by automating the process and improving accuracy. However, success depends on proper preprocessing and parsing of log data. Finally, there's a strong need for models that are not only accurate but also robust and interpretable, so their decisions can be trusted and understood.

This internship provided valuable exposure to both practical and research aspects of log-based anomaly detection, reinforcing the cooperation between the **École Supérieure en Informatique de Sidi Bel Abbès** and **Sapienza University of Rome**. The skills and connections developed during this period will directly benefit my research and the supervision of student projects at ESI-SBA.

Looking ahead, several promising directions can enhance log-based anomaly detection. First, using large language models like Transformers can help better understand log patterns. Second, combining deep learning with clustering techniques may improve accuracy and make the detection process more robust. Lastly, enabling real-time anomaly detection is essential for faster system response and improved reliability.

References

[1] Peter Bodik, Moises Goldszmidt, Armando Fox, Dawn B Woodard, and Hans Andersen. Fingerprinting the datacenter: automated classification of performance crises. In *Proceedings of the 5th European conference on Computer systems*, pages 111–124, 2010.

[2] Mike Chen, Alice X Zheng, Jim Lloyd, Michael I Jordan, and Eric Brewer. Failure diagnosis using decision trees. In *International Conference on Autonomic Computing, 2004. Proceedings.*, pages 36–43. IEEE, 2004.

[3] Zhuangbin Chen, Jinyang Liu, Wenwei Gu, Yuxin Su, and Michael R Lyu. Experience report: Deep learning-based system log analysis for anomaly detection. *arXiv preprint arXiv:2107.05908*, 2021.

[4] Amir Farzad and T Aaron Gulliver. Unsupervised log message anomaly detection. *ICT Express*, 6(3):229–237, 2020.

[5] Shilin He, Jieming Zhu, Pinjia He, and Michael R Lyu. Experience report: System log analysis for anomaly detection. In *2016 IEEE 27th international symposium on software reliability engineering (ISSRE)*, pages 207–218. IEEE, 2016.

[6] Dong Huang, Zhang Yi, and Xiaorong Pu. A new incremental pca algorithm with application to visual learning and recognition. *Neural Processing Letters*, 30:171–185, 2009.

[7] Md Tahmid Rahman Laskar, Jimmy Xiangji Huang, Vladan Smetana, Chris Stewart, Kees Pouw, Aijun An, Stephen Chan, and Lei Liu. Extending isolation forest for anomaly detection in big data via k-means. *ACM Transactions on Cyber-Physical Systems (TCPS)*, 5(4):1–26, 2021.

[8] Yinglung Liang, Yanyong Zhang, Hui Xiong, and Ramendra Sahoo. Failure prediction in ibm bluegene/l event logs. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pages 583–588. IEEE, 2007.

[9] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation-based anomaly detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 6(1):1–39, 2012.

[10] Weibin Meng, Ying Liu, Yichen Zhu, Shenglin Zhang, Dan Pei, Yuqing Liu, Yihao Chen, Ruizhi Zhang, Shimin Tao, Pei Sun, et al. Loganomaly: Unsupervised detection of sequential and quantitative anomalies in unstructured logs. In *IJCAI*, volume 19, pages 4739–4745, 2019.

[11] Wei Xu, Ling Huang, Armando Fox, David Patterson, and Michael I Jordan. Detecting large-scale system problems by mining console logs. In *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles*, pages 117–132, 2009.

[12] Jieming Zhu, Shilin He, Pinjia He, Jinyang Liu, and Michael R Lyu. Loghub: A large collection of system log datasets for ai-driven log analytics. In *2023 IEEE 34th International Symposium on Software Reliability Engineering (ISSRE)*, pages 355–366. IEEE, 2023.

Mohammed Bekkouche
LabRI-SBA Lab., École Supérieure en Informatique de Sidi Bel Abbès (Algeria)