Proposal Bachelor's Thesis

Development and Evaluation of a Hybrid Approach for Automated Error Detection and Classification in LoRaWAN-Based IoT Data Pipelines Using Log Data

Emir Hamulic

University of Applied Sciences Technikum Wien

Student ID: wi23b168
Study Program: B.Sc. Business Informatics
Advisor: Rohatsch Lukas, MSc
Version: 0.1

Date: January 18, 2026

1. Problem Area

1.1. Operational Context

LoRaWAN-based IoT deployments are typically operated as distributed, multi-component systems. End devices communicate with gateways, which forward packets to a LoRaWAN Network Server (LNS). The LNS then routes data to application components or ETL-Pipelines which process data into structured data for databases or dashboards. This "star-of-stars" topology and the central role of the Network Server are core characteristics of LoRaWAN networks. [1]

1.2. Log Volume and Monitoring Limit

Modern IT and IoT backends generate very large volumes of logs. Manual inspection becomes impractical at scale and is only possible to archive with automation of log analysis and diagnosis. [2]

Practitioner evidence reinforces this: in a large survey on log anomaly detection, practitioners explicitly cite the need to "efficiently analyze large volumes of log data," and many expect tools to handle at least 100,000 logs while still delivering near-real-time results. [3]

1.3. Heterogenity and Missing Structure

Operational logs are typically semi-structured, highly heterogeneous, and produced by many components in an interleaved manner. A widely adopted first step in automated log analysis is log parsing (template extraction) to convert raw messages into structured events.

Research further shows that diverse formatting and log types create significant challenges when applying ML methods directly to real-world logs, and that preprocessing/structuring decisions materially impact downstream analysis quality.[4]

1.4. Missing Prioritization

Raw logs typically provide neither a consistent prioritization scheme nor decision-support artifacts such as confidence estimates, rationale, or action guidance. In practice, this forces operators manually looking through and link alerts, driving up the MTTR and operational cost. [3]

1.5. Rule-Based Detection Systems

Rule-based systems and expert-defined patterns are effective for known signatures and for encoding domain knowledge, but they require ongoing effort and are limited when log messages vary substantially or new patterns appear. Early log anomaly detection often relied on rule-based or pattern-driven approaches, while recent work increasingly applies deep neural networks to learn normal behavior and handle patterns in log data.[5]

1.6. ML-Based Detection

The ML component is primarily used as a mechanism to detect and prioritize *unknown* events. Instead of requiring an explicit signature for each failure, anomaly detection techniques learn a baseline of normal log behavior and flag deviations for operational attention. [5, 6] This is particularly relevant for long-tail failures such as new stack traces, unexpected integration errors, or previously unseen combinations of events.[7]

A Random Forest is a comparatively lightweight classifier for tabular log features. It aggregates many decision trees via majority vote and can output class probability estimates, which makes it suitable for operational classification without the complexity of deep models. [8] In a hybrid pipeline, these probability estimates can be used to implement a reject option for unknown events: if the maximum predicted class probability falls below a threshold, the event is tagged as *Unknown* and routed to aggregation and rule-base enrichment. [8, 9]

To handle unknown events in a hybrid classifier, the ML subsystem should not force every input into one of the known error classes. In open-set recognition, inputs from unseen classes must be rejected or mapped to an "unknown" category based on confidence or risk-aware decision rules. [10]

2. Task Description

The core task of this thesis is to develop a proof-of-concept (PoC) system that automatically analyzes, classifies, and temporally aggregates LoRaWAN pipeline log data. The PoC is implemented in Python and MariaDB is used primarily as a persistence layer for structured, classified, and aggregated events to enable queries for monitoring and reproducible evaluation. The aim of the PoC is to conduct a controlled comparison between three strategies using empirical metrics, under consideration of both classification quality and operational advantages.

2.1. Objective

Manual log monitoring does not scale for high-volume and heterogeneous LoRaWAN pipeline logs and provides limited prioritization for incident events. Rule-based detection is precise for known signatures but struggles with message variability and previously unseen events. This work therefore evaluates whether an ML component and a rules-first hybrid strategy is able to improve automated classification and decision support compared to a rule-only method.

2.2. Target Audience

The target audience are technical stakeholders: IoT/backend engineers, applied Data/ML Engineers and operations/SRE roles, working with LoRaWAN data pipelines. The intended use is to provide a reproducible PoC workflow for structured and aggregated error views, and empirical evidence to select an appropriate detection strategy for specific error classes and operational requirements.

2.3. Expected Outcomes

The outcome of this thesis should be that readers are able to reproduce: (i) how to structure LoRaWAN pipeline logs into machine-processable events, (ii) how to implement a rule-based baseline, (iii) how to train and apply an ML classifier with confidence-based handling of *Unknown* events.

2.4. Work Steps

The work is organized into the following technical steps:

1. **Parsing and Structuring:** Convert raw log lines into structured events for deterministic matching, ML features, and aggregation.

2. **Labeling Strategy:** Define an operational error taxonomy (classes, examples, priorities) and a labeling approach for ML and evaluation.

3. **Rule-Based Baseline:** Implement a deterministic detector that assigns classes and priorities for explicit, stable signatures. *Unknown* events should be routed to a residual category.

4. **ML-Based Classifier:** Implement an ML classifier with engineered features and calculate confidence estimates.

5. **Hybrid Decision Logic:** Implement a rules-first policy including confidence-based handling of *Unknown* events via a reject option.

6. **Temporal Aggregation and Storage:** Aggregate classified events over time windows and store summaries for monitoring in MariaDB (counts per class, affected entities, confidence statistics).

7. **Evaluation Setup:** Define a clear test protocol with two complementary scenarios:

   - *Known-classes test (closed-set):* evaluate how well rule-based, ML-based, and hybrid approaches classify log events into the predefined error classes.
   - *Novelty test (open-set):* create a controlled "unknown" test set by holding out selected classes or log templates during training; evaluate whether the approaches correctly assign these cases to *Unknown* while still classifying the remaining known cases reliably.

8. **Empirical Comparison and Reporting:** Compare rule-based, ML-based, and hybrid approaches with metrics at two levels:

   (a) logline level with precision/recall/F1 and confusion analysis

   (b) time-window/incident level with quality of aggregated summaries

   The novelty test reports *Unknown-Recall* and *False-Reject Rate*. The results are summarized and could be used for guidance on when rules, ML, or a hybrid approach is most suitable.

3. Research Questions / Hypotheses

Main Research Question (MRQ):
How effective is a Python-based proof-of-concept for automated log analysis in LoRaWAN-based IoT data pipelines that enables a reproducible comparison of rule-based, ML-based, and hybrid approaches for error classification and temporal aggregation?

Sub Research Question 1 (SRQ1):
Which practically relevant error and warning classes can be derived from real LoRaWAN pipeline logs and formalized into a consistent class catalogue?

Sub Research Question 2 (SRQ2):
How do rule-based, ML-based, and hybrid approaches differ in measurable performance at log-line and time-window/incident level?

Sub Research Question 3 (SRQ3):
Which error classes are better addressed by deterministic rules and which benefit more from ML-based classification, and which class characteristics explain these differences?

4. Methodology

4.1. Research Design

This thesis follows an engineering-oriented research design combining *prototyping* with *quantitative data analysis* and *literature reviews*. The core artifact is a Python-based proof-of-concept (PoC) that transforms raw LoRaWAN pipeline logs into structured, classified, and temporally aggregated events. The artifact is evaluated empirically and compared across three strategies.

4.2. Methods per Research Question

4.2.1 SRQ1: Error and Warning Classes

Methods: Qualitative content analysis and descriptive statistics.
Log messages are reviewed to derive error and warning categories using inductive coding. The class list is consolidated with definitions, class boundaries, and priorities. Descriptive statistics check relevance and coverage via class frequencies across components and time.
Suitability: SRQ1 builds a consistent taxonomy from logs and verifies that classes occur often enough to support automation.

4.2.2 SRQ2: Performance Comparison

Methods: Controlled comparative evaluation and quantitative measurement.
The three detection strategies are evaluated under a fixed protocol with the same dataset, splits, and metrics. Two classes are used for testing. One consists of known classes, and the other is a novelty test created by withholding selected classes or templates during training. Results are reported at log-line and time-window/incident level.
Suitability: SRQ2 requires objective, comparable results. A controlled setup ensures that any performance differences are due to the detection strategy, not to changes in the data, splits, or evaluation procedure.

4.2.3 SRQ3: Suitability of Rules and ML by Error Class

Methods: Per-class analysis and deductive interpretation.
SRQ2 results are analyzed per error class and related to class characteristics such as stability, variability and rarity. This supports assessing whether rules, ML, or a hybrid approach fits best per class.
Suitability: SRQ3 translates empirical results into guidance by linking performance to characteristics.

5. Kind of expected results

5.1. Expected Project Outputs

Beyond the Research Question results, the thesis is expected to deliver these outputs:

- A technical introduction to LoRaWAN and the observed IoT data pipeline context.
- A technical overview of ETL Pipelines and the advantages of logging.
- A quantitative analysis of real LoRaWAN log data from the enterprise setting.
- A reproducible Python-based PoC for automated log classification and temporal aggregation, storing summaries in MariaDB.
- Structured error overview replacing raw logs.

5.2. Expected Results of the Research Questions

The expected results of this thesis are structured by research question and follow directly from the selected methods and the developed PoC artifact.

For **SRQ1**, the qualitative content analysis of real LoRaWAN pipeline logs is expected to produce a class catalogue that is suitable for automated processing. The catalogue will define a bounded set of error and warning classes with meaningful priorities and clear class definitions. Descriptive statistics are expected to return a baseline of the observed log data, such as class frequencies and distributions over time and components.

For **SRQ2**, the controlled comparative evaluation is expected to lead to a reproducible performance comparison between rule-based, ML-based, and hybrid approaches. This includes a standardized protocol, metric results at log-line level, and monitoring-oriented results at time-window or incident level. The outcome will provide measurable evidence of performance differences, including how confidence-based handling of *Unknown* events affects both novelty detection and the quality on known classes.

For **SRQ3**, the per-class analysis is expected to produce a mapping that explains which error classes are most effectively handled by rules, by ML, or by a hybrid strategy. The result will match observed performance patterns to class characteristics such as stability, variability, context and rarity. This is expected to give guidance for how to prioritize rule development, where ML adds operational value, or how this could be used for *Unknown* cases as input for iterative refinement of the class catalogue and the rule base.

5.3. Overview: Research Questions, Methods, and Expected Result Types

Research questions / hypothesis	**Method(s)**	**Expected kind of result**
SRQ1: Which practically relevant error and warning classes can be derived from real LoRaWAN pipeline logs and formalized into a consistent class catalogue?	Qualitative content analysis; descriptive statistics	A class catalogue with clear definitions, boundaries and priorities including representative log patterns.
SRQ2: How do rule-based, ML-based, and hybrid approaches differ in measurable performance at log-line and time-window/incident level?	Controlled comparative evaluation; quantitative performance measurement	An empirical comparison report including per-class metrics on log-line level and monitoring-oriented evaluation on time-window/incident level for all three approaches.
SRQ3: Which error classes are better addressed by deterministic rules and which benefit more from ML-based classification, and which class characteristics explain these differences?	Per-class analysis; deductive interpretation	An evidence-based suitability mapping of error classes to detection strategy, with explanations based on observable class characteristics.

Table 1: Mapping research questions to methods and expected result types.

6. Timetable

6.1. Milestone Plan

Nr	Name	Plan	Adapted by	Actual Date
M1	Proposal approved	31.01.2026	–	–
M2	Literature review completed	02.03.2026	–	–
M3	Data preparation completed	12.03.2026	–	–
M4	PoC implementation completed	01.04.2026	–	–
M5	Partial submission and presentation	10.04.2026	–	–
M6	Complete first version handed in	01.05.2026	–	–
M7	Final thesis version handed in	22.05.2026	–	–

Table 2: Milestone plan vs. actuals.

6.2. Project Plan (Gantt Chart)

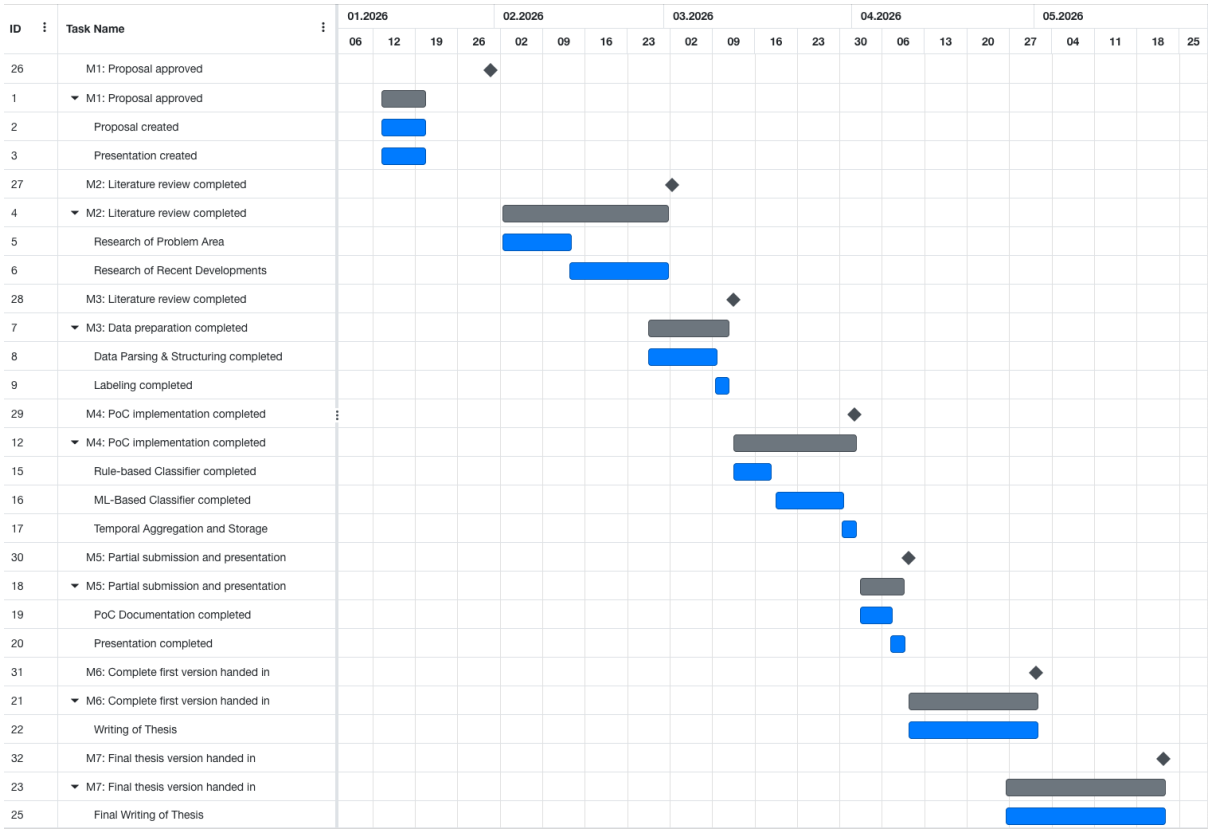

Figure 1: Project plan (Gantt chart).

List of Tables

List of Figures

References

[1] LoRa Alliance, *LoRaWAN® Specification v1.1*, Online technical specification, Accessed 2026-01-17. [Online]. Available: https://resources.lora-alliance.org/technical-specifications/lorawan-specification-v1-1

[2] A. J. Oliner, A. Ganapathi, and W. Xu, "Advances and challenges in log analysis," *Communications of the ACM*, vol. 55, no. 2, 2012. DOI: 10.1145/2076450.2076466 [Online]. Available: https://dl.acm.org/doi/epdf/10.1145/2076450.2076466

[3] X. Ma et al., *Practitioners' expectations on log anomaly detection*, arXiv preprint, 2024. arXiv: 2412.01066 [cs.SE]. [Online]. Available: https://arxiv.org/abs/2412.01066

[4] P. He, J. Zhu, Z. Zheng, and M. R. Lyu, "Drain: An online log parsing approach with fixed depth tree," in *2017 IEEE International Conference on Web Services (ICWS)*, 2017, pp. 33–40. DOI: 10.1109/ICWS.2017.13 [Online]. Available: https://jiemingzhu.github.io/pub/pjhe_icws2017.pdf

[5] M. Landauer, S. Onder, F. Skopik, and M. Wurzenberger, "Deep learning for anomaly detection in log data: A survey," *Machine Learning with Applications*, vol. 12, p. 100 470, 2023. DOI: 10.1016/j.mlwa.2023.100470 [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666827023000233

[6] M. Du, F. Li, G. Zheng, and V. Srikumar, "Deeplog: Anomaly detection and diagnosis from system logs through deep learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS '17)*, 2017, p. 1285. DOI: 10.1145/3133956.3134015

[7] X. Ma et al., *Practitioners' expectations on log anomaly detection*, Dec. 2024. DOI: 10.48550/arXiv.2412.01066 [Online]. Available: https://www.researchgate.net/publication/386375463_Practitioners'_Expectations_on_Log_Anomaly_Detection

[8] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001. DOI: 10.1023/A:1010933404324 [Online]. Available: https://link.springer.com/article/10.1023/A:1010933404324

[9] S. Vaze, K. Han, A. Vedaldi, and A. Zisserman, "Open-set recognition: A good closed-set classifier is all you need?" In *International Conference on Learning Representations (ICLR)*, 2022. [Online]. Available: https://arxiv.org/abs/2110.06207

[10] W. J. Scheirer, L. P. Jain, and T. E. Boult, "Probability models for open set recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 11, pp. 2317–2324, 2014. DOI: 10.1109/TPAMI.2014.2321392 [Online]. Available: https://ieeexplore.ieee.org/document/6809169