

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 usually 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 therefore log analysis and diagnosis must be automated. [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. Heterogeneity and Missing Structure

Operational logs are produced by numerous components in an interleaved fashion, are highly heterogeneous and are semi-structured. Automated log analysis typically begins by extracting templates that convert raw, unstructured log messages into a structured event representation. Research further demonstrates that diverse formatting and log types create significant challenges when applying machine learning methods directly to real-world logs, and that preprocessing decisions materially impact analysis quality.[4]

1.4. Missing Prioritization

Raw logs usually do not provide a prioritization scheme or support for decisions like confidence estimates, rationale, or action guidance. This forces operators to manually inspect and link alerts, which is driving up the MTTR and operational cost. [3]

1.5. Rule-Based Detection Systems

Rule-based systems are effective for known patterns, but they require effort and are limited when log messages are variable or completely new patterns appear. Initial approaches to log anomaly detection relied mainly on manually defined rules. More recent studies favor deep neural networks that model normal log behavior and identify deviations.[5]

1.6. Machine Learning-Based Detection

The Machine Learning component is primarily used as a mechanism to detect and prioritize *unknown* events. Anomaly detection methods identify outliers for operational attention by learning a baseline of typical log behavior rather than requiring an explicit signature for every failure. [5, 6] This is relevant for long-tail failures such as new stack traces, unexpected integration errors, or previously unseen combinations of events.[3]

A Random Forest is a comparatively lightweight classifier for tabular log features. It is suitable for classification without the complexity of deep models due to the aggregation of many decision trees, which can output estimates for class probabilities.[7] In a hybrid pipeline, these probability estimates can be used to implement a reject option for unknown events. When the maximum predicted class probability falls below a threshold, the event is tagged as *Unknown* and routed to aggregation and sometimes with rule-base enrichment. [7, 8]

With an open-set recognition strategy, the Machine Learning subsystem should not force every input to an error class. Inputs from unseen classes must be rejected or mapped to an “unknown” category based on confidence or risk-aware decision rules. [9]

2. Task Description

The main aim of this thesis is to design and build a prototype system that automatically analyzes, classifies, and temporally aggregates LoRaWAN pipeline log data. The prototype system is implemented in Python. MariaDB is used primarily as a persistence layer for events to enable queries for monitoring and reproducible evaluation. The aim of the prototype 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 if an Machine Learning 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 prototype 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 Machine Learning 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:** Convertation of raw log-lines into structured events for deterministic matching, Machine Learning features, and aggregation.
2. **Labeling:** Definition of an operational error taxonomy (classes, examples, priorities) and a labeling approach for Machine Learning and evaluation.

3. **Rule-Based Classifier:** Implementation of a deterministic detector that assigns classes and priorities for explicit, stable signatures. *Unknown* events should be routed to a residual category.
4. **Machine Learning-Based Classifier:** Implementation of an Machine Learning-based classifier with engineered features and calculated confidence estimates.
5. **Hybrid Classifier:** Implementation of a rule-based Classifier first and a Machine Learning-based classifier classifying based of confidence for *Unknown* events.
6. **Aggregation and Storage:** Aggregation of events over time windows and storing summaries like confidence statistics or affected entities inside MariaDB.
7. **Evaluation:** Definition of a test protocol with two scenarios:
 - *Known-classes test (closed-set)*: Evaluation of how well rule-based, Machine Learning-based, and hybrid approaches classify log events into the predefined error classes.
 - *Novelty test (open-set)*: Creation of a “unknown” test set by holding out classes or log templates during training. Evaluation of whether the approaches correctly assign to *Unknown* while still classifying the remaining known cases.
8. **Comparison and Reporting:** Comparison of rule-based, Machine Learning-based, and hybrid approaches with metrics at two levels:
 - (a) log-line 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, Machine Learning, or a hybrid approach is most suitable.

3. Research Questions

Main Research Question (MRQ):

How effective is a hybrid approach for automated error detection, classification and aggregation in LoRaWAN-based IoT data pipelines?

Sub Research Question 1 (SRQ1):

Which practically relevant error and warning events can be identified from real LoRaWAN pipeline logs and formalized into a consistent class catalogue?

Sub Research Question 2 (SRQ2):

How do rule-based, Machine Learning-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, which benefit more from Machine Learning-based classification, and which 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 prototype 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 explicit definitions, class boundaries, and priorities. Descriptive statistics check the occurrence of classes across components and time.

Suitability: SRQ1 builds a taxonomy from logs and checks the occurrence of classes to enable 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. The novelty test is performed in the open-set setting, where a set of classes or log templates is held out in the training stage, and the methods are evaluated based on how well they are able to classify the held-out samples into the *Unknown* class while maintaining a strong classification performance on the known classes.[9] 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 Machine Learning by Error Class

Methods: Class analysis and interpretation.

SRQ2 results are analyzed per error class and related to class characteristics such as stability, variability and rarity. This supports assessing whether rules, Machine Learning, or a hybrid approach fits best per class.

Suitability: SRQ3 takes empirical results to guidance because it is linking the strategy performance to the strategy 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 reproducible Python prototype for automated log classification and temporal aggregation, storing the events and 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 prototype 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 establish a baseline of the observed log data, such as class frequencies and distributions over time and across components.

For **SRQ2**, a controlled comparative assessment is expected to provide a reproducible performance comparison between all approaches. This includes a standardized protocol, metric results at the 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 whole quality for known classes.

For **SRQ3**, the class analysis produces a mapping that explains which error classes are most effectively handled by the three strategies. The result will relate observed performance patterns to class characteristics such as stability, variability, context and rarity. This is expected to offer practiceable guidance on how to prioritize rule development, where Machine Learning 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, Machine Learning-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 Machine Learning-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)

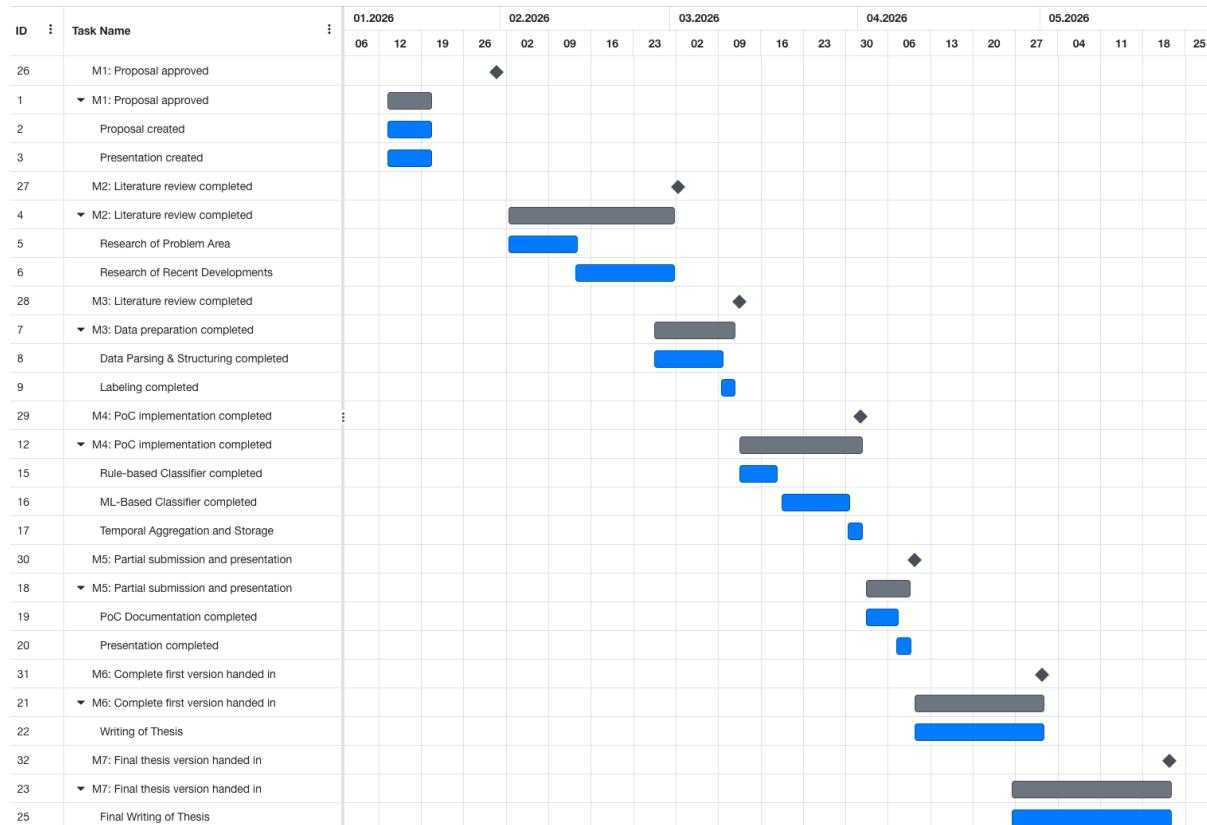


Figure 1: Project plan (Gantt chart).

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. DOI: 10.48550/arXiv.2412.01066 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. 100470, 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] 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>
- [8] 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>
- [9] 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>