# An End-to-end Smart Radio Log Anomaly Detection Architecture for 5G/6G Networks

Marios Avgeris *Member, IEEE,*
Institute of Informatics, University of Amsterdam, Amsterdam, The Netherlands
*m.avgeris@uva.nl*

Aroosa Hameed, Brian Le,
Ericsson Canada
{*aroosa.hameed, brian.le*}*@ericsson.com*

Aris Leivadeas *Senior Member, IEEE,*
Department of Software and IT Engineering, École de technologie supérieure, Montreal, Canada
*aris.leivadeas@etsmtl.ca*

Ioannis Lambadaris *Member, IEEE,*
Department of Systems and Computer Engineering, Carleton University, Ottawa, Canada
*ioannis@sce.carleton.ca*

***Abstract*—In the era of 5G/6G networking, the complexity and scale of modern cellular networks have increased significantly. Consequently, the volume of generated logs has also multiplied, making comprehensive anomaly detection a cumbersome task for operators. In response, this article presents the Radio Log Anomaly Detection (RLAD) architecture, an MLOps-driven pipeline designed to enable automated, end-to-end anomaly detection in log data from next-generation networks. The key contribution lies in a design that integrates continuous training, deployment, and performance monitoring mechanisms to mitigate data drift and ensure model relevance in dynamic telecom environments. The architecture also addresses the practical challenges of labeled data scarcity and class imbalance, enabling robust detection even in unsupervised settings. As a proof-of-concept, we instantiate the pipeline with a lightweight LSTM autoencoder enhanced with attention, achieving 98% accuracy, 99% F1-score, and a 99% precision while outperforming a baseline LSTM. The system delivers anomaly insights via a user-friendly interface that supports operator diagnostics and feedback integration.**

## I. Why is a 5G/6G Log Anomaly Detection Architecture Important?

With the emergence of 5G and beyond networking, the complexity and scale of modern cellular infrastructures have multiplied. In this environment, a single hardware anomaly can disrupt the services of millions of users. Therefore, accurate and timely anomaly detection is crucial for mitigating losses for infrastructure providers [1]. To this end, telecom operators commonly monitor the RAN, transport, and core parts of the network, generating logs describing various runtime events at different levels and processes. With the virtualization of 5G/6G components, multiple monitoring and detection functions, can efficiently coexist within a unified, service-based architecture. This architectural flexibility enables seamless integration of telemetry data streams with anomaly detection pipelines. These logs and telemetry data thus become valuable sources for diagnostic tasks, including identifying vulnerabilities, performance anomalies, and predicting failures and errors [2].

As cellular networks have progressed from small to large-scale infrastructures, the log data generated by the network hardware has similarly evolved. Initially consisting of simple sequential values, logs have transformed into complex, unstructured data containing significant amounts of text and numerical information [3]. This data is generated automatically by the operating system and processes running on infrastructure hardware.

The difficulty of log data processing lies in the non-structural characteristics of the data itself and its temporal correlation. Anomaly detection using rule-based algorithms has been proposed to mitigate human error, yet such methodologies still necessitate human intervention. Consequently, active research is underway to develop complementary real-time monitoring-based anomaly detection systems leveraging machine learning, thereby enhancing automation and scalability [4]. For instance, Almodovar et al. [5] employ a fine-tuned BERT-based language model to capture linguistic patterns in logs while Guo et al. [6] introduce a transformer-based model. However, these methods require substantial computational resources due to their high parameter count, memory consumption, and inference latency. In contrast, the authors in [7] propose a framework that employs Machine Learning Operations (MLOps) for streamlining the machine learning anomaly detection lifecycle in next generation networking environments, specifically in an Open5GCore implementation by benchmarking models like LSTM, MLP, and One-Class SVM. Similarly, in [8], the authors investigate an automated log anomaly detection workflow based on machine learning and natural language processing such as autoencoders and Self-Organizing Maps (SOM), for identifying root causes of failures and vulnerabilities. The evaluation here is performed

among eight widely adopted anomaly detection models.

Despite the tremendous value buried in cellular network logs, how to analyze them effectively remains a great challenge. We mainly identify the following challenges:

1) *Template Identification and Extraction*: As every other log type, cellular logs are a sequence of events produced by hardware and software processes. As such, each log message consists of a combination of semantic and numerical information. The challenge here is to identify the constant semantic part, i.e., the *template* of the message, and extract the variable, usually numerical part which differs with each logged event. Then, properly trained anomaly detection models can capture temporal dependencies in sequences of these two parts and detect anomalies based on unexpected patterns. However, the diverse nature of cellular logs that are generated across multiple components and processes with varying formats, further complicates the accurate extraction of templates and requires advanced methods to handle inconsistencies across diverse log processes.
2) *Dataset/Class Imbalance*: In 5G/6G infrastructures, class imbalance is a prominent challenge due to the inherent characteristics of these systems. Due to the massive scale of deployed components and the continuous, high-volume generation of cellular logs, true anomalies or failures are relatively rare/underrepresented events. Modern network components are designed to be highly resilient and self-healing, which results in a natural scarcity of reported or observable failure instances. Consequently, datasets are often dominated by logs representing normal behavior, leading to a significant skew in class distribution. This imbalance hinders the performance of anomaly detection models by making them overly sensitive to the dominant (normal) class. Moreover, it can impair the model's ability to generalize to unseen or emerging anomalies, which is particularly important in dynamic and evolving environments such as 5G/6G networks.
3) *Labeled Data Scarcity*: In legacy industries such as telecommunications, the incentive towards log data labeling has been low until the recent machine-learning-induced disruption. Still, the relatively high cost and tedious labor required, makes obtaining labels for 5G/6G logs a challenging task that constitutes the training and use of supervised anomaly detection models impractical.
4) *Data Drift*: As network components are upgraded or replaced, whether at the hardware, firmware, or software level, log formats may change, new log features may be introduced, and existing ones may be deprecated. These changes result in a shift in the statistical properties of the data over time, leading to a mismatch between the data distributions used during model training and those observed in production. Such discrepancies degrade the performance of trained anomaly detection models, which rely on the assumption that input data during inference resembles the training data. Without mechanisms for adaptation, this drift can significantly hinder a model's sensitivity to emerging or evolving anomalous behaviors.

To overcome these challenges, in this work we propose RLAD, an end-to-end smart Radio Log Anomaly Detection architecture for next generation networks. To the best of our knowledge, RLAD is the first systematic architecture proposed for log-based anomaly detection in cellular networks that will allow troubleshooters in the telecommunications area to work rapidly and with less system knowledge. Specifically, we make the following twofold contribution:

- We propose the RLAD architecture as the first end-to-end cellular log anomaly detection framework specifically designed around an MLOps-based pipeline [9]. Its novelty lies in systematically automating model lifecycle management through *Continuous Training* and integrated performance monitoring, enabling immediate response to data drift and real-time retraining, triggered by observed performance degradation and operator feedback. Additionally, our architecture facilitates Continuous Integration/Continuous Delivery (CI/CD) operations, ensuring that its components and models remain up to date with the latest insights from independent controlled experiments. In this way, we reduce the need for manual intervention, but also operational costs associated with maintaining model relevance in the highly dynamic networking field.
- To overcome labeled data scarcity and imbalance, we advocate for the adoption of unsupervised machine learning models for the semantic based anomaly detection. We modify off-the-shelf log parsers to automate the separation of the semantic and the numerical part of the messages. As an instantiation of the proposed architecture, we employ LSTM and attention-based autoencoder models to analyze the log message sequences from different hardware entities. This choice illustrates how RLAD can support models that capture latent temporal and structural patterns in real-world log templates, while maintaining operational efficiency.

The rest of the paper is structured as follows. Section II presents the proposed RLAD Architecture. Section III demonstrates the efficiency of the architecture through a proof-of-concept case study. Section IV provides few future directions and open challenges, while Section V concludes the paper.

## II. The RLAD Architecture

An overview of the proposed RLAD architecture is illustrated in Fig 1. To accommodate the end-to-end automated anomaly detection in 5G and beyond network infrastructures, we design this architecture inspired by the MLOps concept; unlike traditional anomaly detection approaches, which typically involve sporadic updates and manual retraining, RLAD continuously monitors performance drift and automatically retrains and redeploys anomaly detection models in real-time. To achieve this, the proposed architecture is structured into five distinct functional layers, each facilitating the process flow from data generation to delivery to telecom operators.

### A. Log Collection

Next-generation networks comprise multiple subsystems, components, and interfaces, each generating heterogeneous

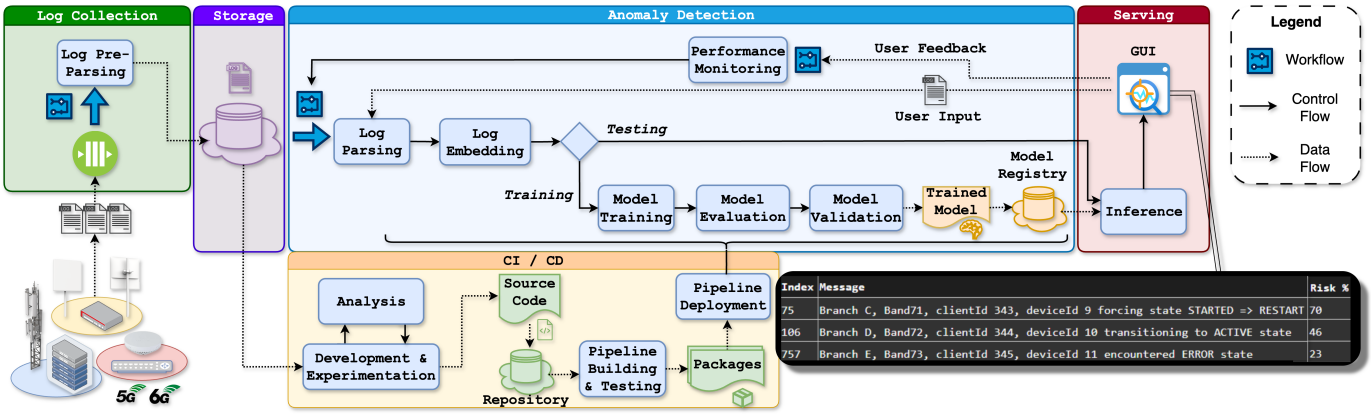


Fig. 1. RLAD Architecture Overview

and voluminous log data structured in time-series format. Cellular logs include automatically generated sequential measurements from hardware processes, such as antenna power, signal strength, and synchronization delays. We use two ways of classifying them depending on their nature: *i) semantic/linguistic data* and *ii) numerical data*. Usually, in a single log message we find data belonging to either or both of these two classes. Specifically, we typically encounter descriptive information, error messages, event descriptions, and contextual details mixed with timestamps, error codes and numerical metrics. Combining both data types improves anomaly detection robustness by leveraging their complementary characteristics.

Due to dynamic data processing demands, log collection components must be flexible and scalable. On the client side, distributed ingestion agents collect logs from 5G/6G network elements (e.g., gNB, network devices, COTS servers) and publish events to a central streaming platform. A workflow orchestration component polls the streaming platform for specific events (e.g., new log data arrival from specific source). This component subsequently triggers the execution of a workflow that extracts the data, pre-parses them in a format appropriate for the `Storage` solution and then stores the logs. This `Log Pre-Parsing` component serves as a preparatory step for storing the log data in a way that optimizes the subsequent anomaly detection analysis. During this process, the collected data are cleaned and formatted by standardizing their structure to ensure consistency. This includes the designation of key attributes regarding the source of the data such as radio types, device releases, software versions and product numbers.

## B. Storage

We utilize cloud-based storage to meet scalability and flexibility requirements by dynamically adjusting storage for high-volume logs. An integrated analytics and search platform indexes and retrieves data efficiently; these capabilities not only support the smooth flow of data to the `Log Anomaly Detection` component but also facilitate controlled experimentation and validation for maintaining and updating the framework's source code (`CI/CD`). Lifecycle management policies are also configured to manage data effectively and ensure that older data are appropriately archived or purged in alignment with compliance and operational efficiencies. On the one hand this reduces the overhead for managing the physical infrastructure and on the other it aligns with the elastic and distributed nature of modern infrastructures.

## C. Anomaly Detection

The `Log Parsing` component initiates the anomaly detection pipeline and can be triggered either by operator input (for *testing*) or by the `Performance Monitoring` automated workflow (for *training*). Accordingly, its data input is a log file, i.e., a collection of time-series cellular log data for analysis, or a batch of data from the `Storage`. Each invocation of the `Log Parsing` component is parameterized by the characteristics of the source of the log data, as well as the model that the data are going to train or be tested against. It parses stored log data, extracting numerical and semantic features and structuring them into log templates (see Fig. 2) A log template is the constant part of the log message that does not depend on the internal parameters that the system is considering at the time of generating the message. Complementary to this, logs are sorted chronologically, grouped by generating processes, and split into sessions. The subsequent `Log Embedding` captures semantic relationships by vectorizing logs in a high-dimensional space for analysis.

If the workflow is initiated for training, the `Model Training` component is triggered. In our implementation we leverage the unique capabilities of LSTM and attention-based autoencoders to handle semantic and sequential patterns

**Raw Logs**

```
2023-06-09 04:26:35.09 INFO tDProc INJECT_REQ_START; sRef:308,
cRef:8195, dId:3, iId:0x1
2023-06-09 04:26:35.48 INFO rTrDServer received msg: 0x17060,
isender: 0x2c008
```

**Parsed Logs**

| Datetime | Level | Process | Log Template | Parameters |
|---|---|---|---|---|
| 2023-06-09 04:26:35.09 | INFO | tDProc | INJECT_REQ_START; sRef:<*>, cRef:<*>, dId:<*>, iId:<*> | 308, 8195, 3, 0x1 |
| 2023-06-09 04:26:35.48 | INFO | rTrDServer | received msg: <*>, *isender:* <*> | 0x17060, 0x2c008 |

Fig. 2. Log Parsing Example

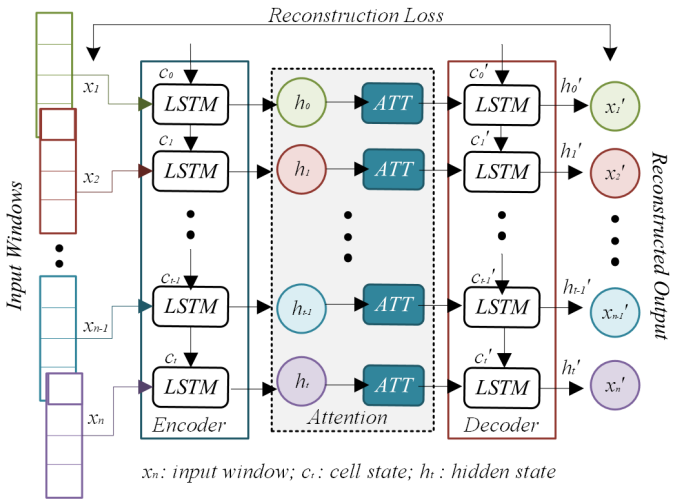


Fig. 3. LSTM-Attention -based Autoencoder Model for Anomaly Detection

within the hardware log data, as shown in Fig. 3. This approach is effective for log analysis because LSTM captures the temporal dependencies inherent in sequential log data, and the attention mechanism further enhances learning by enabling the model to focus on the most relevant parts of the input log sequence, improving its ability to detect subtle anomalies. The model effectively reconstructs normal log sequences and flags deviations, making it powerful for unsupervised anomaly detection in log data without the need for labeled training data. During training, estimators are fitted to model the time series data. During inference, the anomaly score is calculated through the difference between the predicted and actual value for that time point. To achieve higher accuracy, we maintain various versions of the autoencoder models, specifically trained on log datasets that correspond to the various versions of the cellular hardware/software. In this way, we ensure that the anomaly detection process is tailored to the specific equipment characteristics. Additionally, our *Continuous Training* approach ensures models remain updated with new data, equipment versions, and network configurations; it is triggered automatically by new data collected in the `Log Storage`, performance drift reported by the `Performance Monitoring` component, or code updates (`CI/CD` pipeline). We note that the architecture remains agnostic to the specific model used, allowing for future integration of more advanced or specialized learning techniques as needed.

The newly trained models are evaluated by the `Model Evaluation` component using performance metrics tailored to their specific roles. A threshold for anomaly detection is determined based on the 99th percentile of reconstruction errors from both training and test data, allowing the model to identify deviations from normal behavior. Next the model reconstructs the test data, and the reconstruction errors are calculated. Instances with reconstruction errors above the threshold are classified as anomalies. The evaluation is done in such a way that predictions from the model are compared against ground truth labels (anomalies or normal), and metrics such as accuracy, precision, recall, and F-score are calculated to assess the performance of the anomaly detection model. Following the evaluation, the `Model Validation` component verifies that performance metrics meet pre-defined acceptance thresholds for operational deployment. It also checks for overfitting, compatibility with deployment environments, and consistency across retraining iterations before the model is registered. The `Model Registry`, a cloud-based solution, stores model versions alongside their training environments and dependencies, ensuring traceability and reproducibility.

Finally, the `Performance Monitoring` component continuously tracks key metrics such as model reconstruction error trends, inference latency, and resource utilization. Performance degradation is flagged when the reconstruction error distribution for incoming logs shifts significantly compared to the model's baseline (e.g., a sustained increase in the 99th percentile of reconstruction errors). In addition, drops in precision/recall on feedback-labeled data, or a significant divergence between training-time and serving-time behavior (training-serving skew), also act as triggers. These alerts initiate automatic retraining workflows or notify operators for manual inspection. Operator feedback via the `GUI` is also incorporated as validation input for monitoring model drift.

### D. Serving

The `Inference` component applies trained autoencoders to analyze incoming logs. It selects the appropriate model version, calculates reconstruction errors for subprocesses, and compares these errors against thresholds. Instances exceeding thresholds are flagged as anomalies; otherwise, they are classified as normal. RLAD also provides a `GUI` for telecommunication operators to easily interact with the anomaly detection process. Operators upload log files and specify essential metadata, such as hardware/software versions and equipment type. Through the `Inference` component the appropriate model from the `Model Registry` is selected, the logs are processed, and the `GUI` visually displays anomaly scores, highlighting potential issues clearly. It also generates a downloadable report summarizing detected anomalies and recommended actions. Additionally, operators can provide feedback directly through the `GUI` to enhance future anomaly detection accuracy.

### E. Continuous Integration & Continuous Deployment (CI/CD)

The `CI/CD` pipeline is a critical component that ensures the seamless integration, testing, and deployment of the log anomaly detection architecture components. Here, the `Development & Experimentation` component is responsible for the continuous refinement and testing of new anomaly detection approaches. It supports iterative experimentation by allowing researchers to prototype updates to the `Anomaly Detection` pipeline in a controlled environment. Once these updates pass initial validation, the `Analysis` component evaluates their effectiveness through quantitative and qualitative assessments, including performance, resource efficiency, and deployment impact. If results are satisfactory, the corresponding source code is committed to a version-controlled repository for traceability and collaboration. Then,

the `Pipeline Building & Testing` component performs automated testing processes to verify the the correctness and effectiveness of the updated source code. Tests ensure compatibility and verify performance improvements, preventing regressions and degradation in detection capabilities. After successful testing, pipeline components are packaged into deployable software artifacts, ensuring consistency and simplifying the deployment process. Finally, the prepared packages are automatically deployed into the production environment through the `Pipeline Deployment` component. This is an automated process, ensuring quick updates, efficient scaling, and minimal manual intervention. Continuous monitoring and feedback loops then restart the `CI/CD` cycle, promoting agility and robustness in the RLAD architecture.

## III. Case Study: Semantic Radio Anomaly Detection

We hereby conduct a preliminary implementation and evaluation of the RLAD architecture, as envisaged in this work. The whole infrastructure was deployed on a proprietary Ericsson Kubernetes cluster. For publishing the availability of new log data from the client side, the Apache Kafka distributed messaging system was used. To orchestrate the triggers of `Log Collection, Storage, Anomaly Detection` and `CI/CD` pipelines, the open-source container-native workflow engine *Argo* was utilized. Specifically for the `CI/CD` pipeline, GitLab was used as the source code repository solution while the packages were managed through *JFrog*. The `Storage` was implemented as an AWS S3 instance with an integrated *OpenSearch* platform for searching, aggregating, viewing, and analyzing the log data in a human friendly way whenever needed. The `GUI` was implemented using JavaScript and HTML/CSS. The rest of the architecture components were implemented using in-house solutions.

Building on deployed components, we now walk through each stage of the RLAD pipeline to highlight how the logs flow through the system from ingestion to delivery of anomaly insight. The pipeline starts with real-time log streaming from 5G radio equipment into the Apache Kafka message bus, initiating the flow from raw data to anomaly detection. Since raw log lines are unstructured and difficult to analyze directly, `Log Parsing` invokes Drain [10], which is a rule-based parser that converts logs into structured templates. Drain builds a parse tree where each path represents a log pattern, matching token-by-token incoming log lines. If no match is found, a new template is created. To further enhance this process, we apply time-process windowing, grouping events from the same process into one-second intervals to capture inter-process relationships and temporal structure. The parsed logs then undergo `Log Embedding`, where they are converted into fixed-length sequences of 12 tokenized events. These sequences are then passed to `Inference` component, which uses an LSTM-based autoencoder enhanced with an attention mechanism. The model is trained on normal logs collected from $500$ historical files recorded in $2024$, covering 5G processes such as Transmit Linearization (TXL), Radio Interface Control Remote (RICR), Antenna Module (AntMod), Transmit (TX) and Receive (RX) process. Each file contained $5000 - 30000$ log lines that are positive samples (logs without anomalies). Training is performed over $100$ epochs with early stopping (patience = 3), using the Adam optimizer with a learning rate of $0.0001$ and a batch size of $64$. The architecture includes two LSTM layers for encoding and decoding sequences, both with input and output dimensions of $12$.

During inference, the model flags sequences with high reconstruction errors as anomalies, indicating deviations from normal patterns. In our study, such anomalies reflect real-world issues frequently observed in 5G radio logs, including configuration mismatches, hardware malfunctions, signal transmission errors, and unexpected resets, especially within the RICR process. These events often manifest as unusual log sequences or abrupt terminations of expected patterns, affecting approximately 2–4% of all sequences. Although infrequent, they are critical to detect early, as they can lead to service degradation or even system failure. To assist operators in responding promptly, detected anomalies are displayed in a `GUI` with timestamps, affected subsystems, and surrounding log context. This interactive interface enables engineers to investigate irregularities quickly and take corrective action.

Table I provides the performance of the proposed LSTM and attention-based model (*LSTM_ATT*) and the baseline *LSTM* [11] model from the literature. We chose LSTM with attention for its balance of efficiency and temporal modeling, making it suitable for real-time telecom environments. While models like Transformers or CNNs show promise, they either require more resources or lack temporal sensitivity. Our evaluation focuses on a baseline LSTM to isolate the contribution of the attention mechanism and align with existing MLOps-based log analysis pipelines, where direct model-level comparison is often impractical due to differing scopes.

Our model is capable of achieving 98% accuracy, 99% precision, 98% recall and 99% F1 score compared to the simple LSTM model. The attention mechanism contributes significantly to the improved performance. Radio logs exhibit temporal dependencies where anomalies are related to specific sequences of events over time. While LSTM models are effective at capturing these dependencies, they can struggle with long-term dependencies due to the vanishing gradient problem. The attention mechanism overcomes this limitation by explicitly capturing both short and long-term dependencies more effectively. We also plot the confusion matrix for the autoencoder-based methods in Fig. 4. The left subplot shows LSTM_ATT and the right plot depicts the simple LSTM model. These results are shown for the RICR process and anomalies are detected at the subprocess level that contain the problematic anomalous logs within the RICR process. The confusion matrix helps us assess how accurately our model classifies the subprocesses as normal or abnormal. We see that the LSTM_ATT model outperforms the simple LSTM model as evidenced by the low false positives and negatives.

Regarding computational requirements, training the LSTM with attention model for 10 epochs on 500 logs takes $\approx 200s$ on an RTX 4060 GPU (8 GB RAM). Once trained, inference on a full test set takes $\approx 6s$, which is a slightly increased compared to the baseline LSTM (with a 88.6% accuracy and

TABLE I
PERFORMANCE EVALUATION OF AUTOENCODERS BASED ANOMALY DETECTION MODELS

| Models | Accuracy | Precision | Recall | F1-score |
|---|---|---|---|---|
| LSTM_ATT | 0.9888 | 0.9943 | 0.9875 | 0.9907 |
| LSTM [11] | 0.8856 | 0.9880 | 0.8206 | 0.8965 |

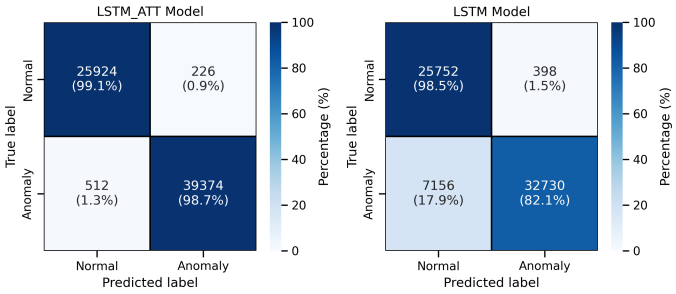


Fig. 4. Confusion Matrix of Autoencoder-based Anomaly Detection Models

89.6% F1). However, the significantly improved accuracy and robustness to complex log sequences justifies the slight computational overhead, while not affecting its suitability for periodic retraining and integration into near real-time monitoring.

We acknowledge that evaluating beyond model performance is important. Scalability and computational efficiency are key for the RLAD deployment. Its horizontally scalable design (Kafka, Argo, AWS S3/OpenSearch) supports high throughput via resource scaling and parallel processing. RLAD's modularity allows independent replacement of components (e.g., Drain parser, anomaly detection model) and enables extension to multi-process and multi-node setups without pipeline modifications. Furthermore, the system allows for asynchronous inference and model updates, facilitating continuous learning and deployment in dynamic network environments. This architecture makes the system adaptable to the demands of large-scale deployments, where telecom equipment can generate millions of log lines per second. Although a detailed ablation study is not our scope, it represents valuable future work for validating the system's adaptability.

## IV. APPLICATIONS, FUTURE DIRECTIONS & OPEN CHALLENGES

### A. Applications

RLAD's design enables its deployment across a wide range of real-world 5G/6G scenarios where automated anomaly detection is critical for security, performance, and reliability. Below, we highlight representative use cases:

*a) Cybersecurity:* RLAD supports zero trust network management by continuously learning what constitutes "normal" behavior and autonomously flagging subtle deviations, including those arising from AI-driven or peer-to-peer attacks in distributed 6G architectures.

*b) Performance Optimization:* In smart city infrastructures, RLAD can integrate with joint communication and sensing (JCAS) components for 6G networks [12], identifying propagation anomalies, such as obstructions, enabling corrective actions like beam steering or dynamic link switching.

*c) Reliability & Maintenance:* RLAD can be integrated into 5G Radio Dot [13] IIoT systems to enable predictive maintenance by detecting early signs of malfunction or degradation, reducing downtime and enhancing safety.

### B. Future Directions

*a) Edge Inference:* To implement the above scenarios, RLAD needs to be streamlined for the the inference part to be deployed at the network Edge, where feasible. This research will involve exploring Edge AI approaches, i.e., deploying AI algorithms and models on edge, resource-constrained devices.

*b) Real-time Operations:* Another interesting direction involves combining in-switch and in-controller anomaly detection, in an in-network ML fashion (e.g., P4-programmable data planes, eBPF) and utilizing specialized hardware accelerators (FPGAs, ASICs, SmartNICs) to enhance the RLAD model's real time inference capabilities with data-plane-native implementations, enabling low-latency, in-situ anomaly detection.

*c) Privacy:* To enhance privacy preservation, another research direction could involve extending the current MLOps pipeline into a Federated Learning Operations (FLOps) framework that orchestrates decentralized model training across distributed RAN nodes, allowing the models to be updated from local data at each site without requiring raw data centralization. This will reduce latency and bandwidth usage overall in the network, but also enhance data privacy and security, as sensitive information will remain localized to the clients' premises. Collective knowledge from multiple devices will be aggregated to improve overall model accuracy and robustness.

*d) Autonomous Decision Making:* To automate simple corrective actions for common network anomalies, a Deep Reinforcement Learning (DRL) component could be integrated to enrich the framework with decision-making capabilities.

*e) Digital Twins:* A Network Digital Twin (NDT) fed with real-world network telemetry (i.e., logs) could simulate diverse disruption scenarios and safely train DRL-based remediation strategies under realistic conditions before live deployment [14]. Such strategies can for example include the data plane automatically rerouting traffic before an outage, based on anomaly predictions (i.e., self-healing).

### C. Open Challenges

Besides the benefits and opportunities mentioned above, some vital open challenges are yet to be explored for the RLAD architecture to reach its full potential:

*a) Integration with Existing Network Control Frameworks & Real-Time Constraints:* The RLAD architecture must seamlessly integrate with existing network control frameworks (e.g., O-RAN's near-RT RIC) while meeting strict real-time anomaly detection deadlines. Achieving interoperability via standard interfaces without sacrificing detection latency often requires specialized xApp designs or hardware acceleration.

*b) Zero Trust & Secure Decentralized Detection:* Applying Zero Trust principles in a decentralized RAN anomaly detection context (e.g., FLOps) implies that no network component or xApp can be implicitly trusted. This requires clearly

defined enforcement boundaries and per-component authentication to confine even agentic ML/RL-based xApps to minimal privileges. Enforcing such strict isolation while enabling effective cooperative detection remains an open challenge.

*c) Responsible AI & Architectural Compliance:* Ensuring RLAD aligns with industry frameworks (e.g., the AI-RAN Alliance's reference architecture [15]) and upholds responsible AI design principles is another challenge. This includes incorporating model explainability (XAI) and governance into its design and verifying that the system's behavior adheres to the guidelines for ethical AI-driven RAN practices.

## V. Conclusions

The increasing complexity and scale of 5G/6G networks have necessitated the development of advanced mechanisms for anomaly detection within telecommunications infrastructures. To address this need, we present RLAD, an end-to-end smart Radio Log Anomaly Detection architecture for 5G/6G network infrastructures. The proposed architecture leverages a Machine Learning Operations (MLOps)-based pipeline that ensures continuous training and deployment, thereby mitigating data drift and maintaining model relevance over time. Our primary contribution lies in the design of this automated and operationally integrated pipeline, which supports scalable and real-time log anomaly detection. As a demonstration, we instantiate the architecture with LSTM and attention-based autoencoder models for log analysis, providing an effective solution for unsupervised anomaly detection in sequential log data. Evaluation of our prototype shows that the LSTM with attention model delivers strong anomaly detection performance, achieving 98% accuracy, 99% F1-score, and a 99% precision significantly outperforming the baseline LSTM model. Future work will include exploring the potential use of Large Language Models (LLMs) as an enhanced, linguistics-based log anomaly detection solution for our architecture.

## References


[1] M. Doan and Z. Zhang, "Deep learning in 5g wireless networks - anomaly detections," in *29th Wireless and Optical Communications Conference (WOCC)*, 2020, pp. 1–6.

[2] L. F. Maimó, Á. L. P. Gómez, F. J. G. Clemente, M. G. Pérez, and G. M. Pérez, "A self-adaptive deep learning-based system for anomaly detection in 5g networks," *IEEE Access*, vol. 6, pp. 7700–7712, 2018.

[3] T. Sundqvist, M. Bhuyan, and E. Elmroth, "Robust procedural learning for anomaly detection and observability in 5g ran," *IEEE Transactions on Network and Service Management*, pp. 1–1, 2023.

[4] B. J. Gort, G. M. Kibalya, M. A. Serrano, and A. Antonopoulos, "Forecasting trends in cloud-edge computing: Unleashing the power of attention mechanisms," *IEEE Communications Magazine*, 2024.

[5] C. Almodovar, F. Sabrina, S. Karimi, and S. Azad, "Logfit: Log anomaly detection using fine-tuned language models," *IEEE Transactions on Network and Service Management*, vol. 21, no. 2, pp. 1715–1723, 2024.

[6] H. Guo, J. Yang, J. Liu, J. Bai, B. Wang, Z. Li, T. Zheng, B. Zhang, J. Peng, and Q. Tian, "Logformer: A pre-train and tuning pipeline for log anomaly detection," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 1, 2024, pp. 135–143.

[7] A. Gopikrishnan, A. Rao, A. Prakash, V. Gowtham, F. Schreiner, M. Corici, C. Hein, and T. Magedanz, "Machine learning pipeline for anomaly detection in next generation networks," in *2024 IEEE Conference on Standards for Communications and Networking (CSCN)*. IEEE, 2024, pp. 196–201.

[8] A. C. de Moura, M. F. Caetano, J. J. Gondim, A. Araujo, M. A. Marotta, L. Bondan *et al.*, "Anomaly detection in logs: A comparative analysis of unsupervised algorithms," in *13th Symposium on Languages, Applications and Technologies (SLATE 2024)*. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2024, pp. 12–1.

[9] D. Kreuzberger, N. Kühl, and S. Hirschl, "Machine learning operations (mlops): Overview, definition, and architecture," *IEEE Access*, vol. 11, pp. 31 866–31 879, 2023.

[10] P. He, J. Zhu, Z. Zheng, and M. R. Lyu, "Drain: An online log parsing approach with fixed depth tree," in *IEEE international conference on web services (ICWS)*. IEEE, 2017, pp. 33–40.

[11] S. Siami-Namini, N. Tavakoli, and A. S. Namin, "The performance of lstm and bilstm in forecasting time series," in *IEEE International conference on big data (Big Data)*. IEEE, 2019, pp. 3285–3292.

[12] H. Andersson, "Joint communication and sensing in 6g networks," *Ericsson Blog*, 2021, Accessed on 10.22.2025. [Online]. Available: https://www.ericsson.com/en/blog/2021/10/joint-sensing-and-communication-6g

[13] Interface, CPRI Common Public Radio and over Ethernet, PoE Power, "Connecting the dots: small cells shape up for high-performance indoor radio," *Ericsson Review*, 2014, Accessed on 10.22.2025. [Online]. Available: https://www.ericsson.com/en/reports-and-papers/ericsson-technology-review/articles/connecting-the-dots-small-cells-shape-up-for-high-performance-indoor-radio

[14] P. Öhlén, C. Johnston, H. Olofsson, S. Terrill, and F. Chernogorov, "Network digital twins–outlook and opportunities," *Ericsson Technology Review*, vol. 2022, no. 12, pp. 2–11, 2022.

[15] AI-RAN Alliance, "AI-RAN Alliance Vision and Mission White Paper," Tech. Rep., 2024, Accessed on 10.22.2025. [Online]. Available: https://ai-ran.org/wp-content/uploads/2024/12/AI-RAN_Alliance_Whitepaper.pdf


## Biographies

**Marios Avgeris** (Member, IEEE) received his diploma from the Department of Electrical and Computer Engineering at the National Technical University of Athens in 2016 and his PhD in 2021. He is currently an Assistant Professor with the Informatics Institute, University of Amsterdam, Netherlands.

**Aroosa Hameed** received her Ph.D. degree from École de Technologie Supérieure (ETS), Montreal, Canada, in August 2023. She was a Postdoctoral Fellow with the Department of Systems and Computer Engineering, Carleton University, Ottawa, Canada.

**Brian Le** graduated from University of Ottawa with a BS in Electrical Engineering in 1990. Brian is a senior telecom/real-time application developer and product support specialist at Ericsson, Ottawa, Canada.

**Aris Leivadeas** (Senior Member, IEEE) is a Professor with the Department of Software and Information Technology Engineering at the École de Technologie Supérieure (ETS), Montreal, Canada. He received his Ph.D degree in Electrical and Computer Engineering from the National Technical University of Athens, Greece, in 2015.

**Ioannis Lambadaris** (Member, IEEE) received a Ph.D. degree in Electrical Engineering from the University of Maryland, College Park, MD, USA in 1991. He joined the Department of Systems and Computer Engineering in Carleton University in September 1992. Currently, he is a Chancellor's professor in the same department.