National College of Ireland

AI-Powered IoT Anomaly Detection

MSc AI1B Research Project

AI-Powered IoT Anomaly Detection

Amey Manishkumar Bhangire

Student ID: x23345314

School of Computing

National College of Ireland

Supervisor: Prof. Lavish Thomas

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Amey Manishkumar Bhangire
Student ID:	x23345314
Programme:	MSc in Artificial Intelligence **Year:** 2024-25
Module:	MSc (Research) Practicum/Internship Part 2
Supervisor:	Prof. Lavish Thomas
Submission Due Date:	Monday 11th August 2025 by 2pm
Project Title:	AI-Powered IoT Anomaly Detection
Word Count:	**7552** **Page Count: 22**

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Amey Manishkumar Bhangire

Date: 11-08-2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

AI-Powered IoT Anomaly Detection

Amey Manishkumar Bhangire

x23345314@student.ncirl.ie

Abstract

This project focuses on detecting unusual behavior in IoT sensor data using a mix of simple and smart techniques. With more devices collecting data, it is important to make sure that the data is reliable and not affected by errors or attacks. In this study, we used the Intel Berkeley Lab dataset and added fake errors to test our models. We applied three main methods—Isolation Forest, One-Class SVM, and an Autoencoder—to find these unusual patterns. We also used a reinforcement learning agent to help make better decisions over time. All models were tested using standard accuracy checks like precision, recall, and F1-score. One-Class SVM performed the best, especially in finding real problems. We built the system to run in real time using a simple Streamlit interface, and it can also work on small edge devices like Raspberry Pi. This project offers a smart and useful solution to keep IoT systems safe by finding problems early and taking the right steps quickly.

1 Introduction

1.1 Background of the Study

The number of IoT-oriented devices have been growing explosively in recent years, stimulating the smart system development in the energy, healthcare, logistics, and manufacturing industries, etc. By 2025, the number of connected IoT devices globally had surpassed 17.1 billion, up from 15.1 billion in 2023, amounting to CAGR of 9.1% (Statista, 2025). It keeps monitoring while streaming vast amount of sensor data, often in real time, fostering prediction, automation, and optimization in system level.

But this proliferation of IoT is also driving profound security, reliability, and operational challenges as well. For example, more than 38% of reported incidents to critical infrastructures in 2024 were linked to compromised IoT or industrial control systems (Cybersecurity and Infrastructure Security Agency [CISA], 2024). Furthermore, the average number of anomalies reported on a daily basis in industry networks sensors rose by 27% from 2023 and 2024 (Gartner, 2025). These anomalies can originate from innocuous sources, such as aging hardware or sensor drift, while potentially reflecting more serious phenomena, like cyber intrusions or physical sabotage.

Hikrishnan (2014) stated that the classical anomaly detection methodologies, which are rule-based or based on statistical thresholds, do not perform well due to the dynamic and multidimensional nature of the data observed from sensors. They tend to have high false positive rates, lack scalability, and are not adaptable to environmental changes. One-Class SVMs, ML, especially the unsupervised models such as Isolation Forest, Autoencoders, and

One-Class SVM, has been extensively used for anomaly detection. These models are able to capture more complex patterns among multiple variables without needing labelled anomalies, which makes them especially suitable for sensing setups with limited labelled data.
However, although the ML models improve the anomaly detection, but in general they are passive. That is, they can spot anomalies but not automatically advise or take corrective actions. This gap is particularly significant in mission-critical domains where responsive delay may result in failure, dangers to life or threats to data integrity. Accordingly, there is an increasing demand for mechanisms that are not only purely reactive, but can also make autonomous, adaptive decisions. Reinforcement Learning (RL), which allows agents to learn good actions by interacting with the environment, provides a promising framework to address this gap. It allows systems to not only recognise unexpected value but also decide dynamically on corresponding action.

1.2 Problem Statement

Existing IoT anomaly detection systems mainly concentrate on the principles of classification and alert, but most of them hardly integrate well with decisions and control systems. Supervised and unsupervised ML models have been successful at detecting abnormal behaviour in sensor data, but for the most part, the current implementations do not take action on the anomaly once detected. In practice acting fast -e.g., starting cooling systems when noticing unexpected temperature readings or isolating network segments under attack- is often a requirement. This provides an inevitable lag between detection of toxins and a response to that detection.
Moreover, many works also rely on static thresholding or offline learning that might not generalise effectively to the variations in the observed sensor behaviour over time. These methods are also typically “black boxes” in that it is hard to explain or interpret what the model predicts. Another weakness is that there is no correlation between anomaly detection and RL so as to enable adaptive control mechanisms to learn optimal control policies.

1.3 Research Motivation and Rationale

The increasing dependence on Internet of Things (IoT) infrastructure in applications like smart manufacturing, healthcare, and critical utilities raises the importance of real-time monitoring and responding to sensor anomalies in a high stakes manner. Machine learning has done a great deal to further the state of the art in anomaly detection, yet they do not make autonomous decisions. This absence of intelligent response increases the possibility of system delay, failure or damage and the like.
Reinforcement Learning (RL) provides a promising perspective to narrow this gap, allowing systems to learn how to optimally respond to dynamic malfunctions in online. However, its application within sensor-based environments is not fully explored, particularly in lightweight or resource limited environments.

The work is inspired by the requirement of a self-adaptive control system that can recognise and respond to anomalies using learnt experience. By integrating unsupervised anomaly detection with RL-based action modelling, this work confronts practical and immediate needs--moving farther from detection toward self-response in an intelligent IoT context.

1.4 Research Aim

This work seeks to provide a cohesive solution that can be used for real-time anomaly detection, together with adaptive response in an IoT sensor network by combining ML classifiers and RL control agents.
Research Objectives

The objectives of the study are as follows:

- To pre-process and analyse raw sensor data from real world IoT data and find important properties that can indicate an anomaly.
- Implementation of various machine learning models (Isolation Forest, One-Class SVM, Autoencoder) for unsupervised anomaly detection and comparison.
- To create a user-defined reinforcement learning environment, which models sensor behaviour and trains a DQN-based take action for anomaly response.
- Combining RL-based actions in the case of processed detection outputs into a user-interactive real-time visualisation dashboard with Streamlit.

1.5 Research Questions

- **How can reinforcement learning be integrated with unsupervised anomaly detection models to enable real-time adaptive control in IoT sensor networks?**
- **What level of accuracy and responsiveness can be achieved by combining classical anomaly detection techniques (e.g., Isolation Forest, Autoencoder) with Q-learning for decision-making in temperature-sensitive IoT environments?**

1.6 Research Gap

For instance, EZDEC (Alghamdi et al., 2023), among other algorithms, has been increasingly used to detect anomalies in time series sensor data, and with the wide-spread adoption of IoT devices in smart-grids, health monitoring systems, and industrial automation, the amount as well as the speed of time series sensor data has grown drastically. While traditional machine learning methods, i.e., isolation forests, and one- class SVM are already successfully used for anomaly detection, the models can provide only a passive diagnosis; that is, they can detect the anomalies but do not suggest or take any action to mitigate that condition (Islam et al., 2024). This results in a very real operational bottleneck, particularly for real-time case scenarios, in which an action taking too long to reach a decision and act on it may cause preventable system failure or safety hazard.

Furthermore, the majority of the prior works tackle the detection and control as two separate modules while the reinforcement learning (RL) is commonly used for robotic or gaming tasks, not for the intelligent handling of the sensor anomalies. For example, the work of Gao et al. (2024) applies for HVAC control and energy efficiency in RL, but these systems operate using event triggers that are predefined rather than learning based on unsupervised anomaly detection feedback. The combination of unsupervised detection with real-time decision policies with RL is scarcer especially in environments that include unreliable/hide-dimensional data from sensors.

Moreover, it's difficult to obtain real-world datasets of fine-grained anomaly labels, which makes the supervised methods more complex. Hybrid models are suggested in recent work, but it is frequently heavy for computation without applicability for edge deployment or low-power IoT domains (Li & Chao, 2025). There is poor development of lightweight and scalable frameworks that include interpretable anomaly detection, as well as policy-based control mechanisms that evolve as the system is used.

This paper seeks to bridge this gap by designing an end-to-end system architecture, where unsupervised models detect anomalies and an RL agent, Q-learning based, makes optimal decisions (e.g., turn on the cooling, raise alert). Contribution: By establishing a feedback loop between detection and response, this work contributes to a self-healing IoT control system—a vision that is not fully addressed by existing literature.

2 Literature Review

2.1 Threats in the Medical IoT Systems

The rapid expansion of the Internet of Things (IoT) ecosystem, now projected to surpass 30 billion connected devices globally by the end of 2025 (Statista, 2024), has amplified the need for robust and adaptive anomaly detection mechanisms. These devices often operate in safety-critical environments—smart cities, industrial control systems, and healthcare infrastructure—where faults or cyberattacks can yield severe disruptions. Conventional machine learning methods, although widely adopted, have limitations in environments requiring real-time adaptability, autonomous decision-making, and fault tolerance. This gap has catalysed the integration of Reinforcement Learning (RL) into the IoT anomaly detection pipeline, offering a promising avenue for dynamic, context-aware response systems.

Ryu et al. (2022) critically evaluated traditional ML-based anomaly detection approaches using the Edge-IIoTset dataset and concluded that static thresholding mechanisms often failed under temporal drift and sensor tampering conditions. To address this, they proposed an adaptive deep Q-learning architecture which improved anomaly detection accuracy by 14% in comparison to SVMs and decision trees. Notably, their RL model continuously refined its policy through feedback loops, enabling early detection even in partially labelled scenarios—an inherent challenge in IoT datasets.

Meanwhile, in a comprehensive experimental study, Kaur and Sangal (2023) implemented Proximal Policy Optimisation (PPO) to manage fault-prone IoT sensors in smart grids. Their system not only detected anomalies but also autonomously reconfigured data routing paths to maintain data integrity. The system demonstrated a 27% reduction in system latency and increased detection precision from 82% to 94%. Their findings highlight that RL agents are well-suited for hybrid anomaly-control architectures, particularly when data integrity and system resilience are paramount.

Building upon such adaptive paradigms, Wang et al. (2023) introduced a federated RL-based intrusion detection system, trained across decentralised sensor nodes while preserving privacy. Their approach outperformed centralised models by maintaining high accuracy (92.3%) without violating data residency constraints. This work is significant for real-world applications where legal or organisational policies prohibit centralised data storage, yet collaborative learning is essential.

A different line of inquiry by Adeel et al. (2022) explored the potential of combining deep RL with attention-based neural networks. Their model selectively focused on important sensor sequences, effectively filtering out irrelevant fluctuations. On benchmark datasets like SWaT and WADI, their attention-augmented DQN achieved 96% detection accuracy while reducing false positives by 37%. Their critical insight was the RL agent's ability to learn which features mattered most over time—a crucial capability in volatile industrial environments.

Further insight is provided by the work of Diro and Chilamkurti (2021), who benchmarked actor-critic RL models on real-time water treatment plant data. They demonstrated that RL agents could outperform rule-based expert systems, particularly in handling zero-day sensor failures. The key takeaway was the agent's ability to explore unseen fault patterns and adjust its control strategy on-the-fly—something static algorithms fundamentally cannot achieve.
The integration of RL with edge computing was explored by Singh et al. (2022), who developed an edge-native RL detection framework that minimised cloud dependency. Their system used a lightweight DQN model with pruning techniques, achieving 89% detection accuracy while consuming 40% less power than conventional approaches. This trade-off between computational efficiency and detection performance marks a necessary consideration for resource-constrained IoT environments.

Another novel contribution comes from Ghafir et al. (2022), who tested multi-agent RL (MARL) to manage coordinated responses across distributed sensors. Their study demonstrated that MARL agents outperformed centralised RL in terms of response speed and adaptability, particularly in the presence of cascading faults. The agents, through decentralised negotiation, reduced fault propagation rates by up to 43%.

From a deployment perspective, the study by Akram et al. (2023) addressed real-time adaptability challenges by integrating RL agents with Apache Kafka for stream processing.

Their architecture ensured sub-second response latency and scalability for smart building applications. While implementation complexity increased, the real-time performance benefits validated RL's role in mission-critical settings.

Also relevant is the work of Mahmood and Jalili (2022), who evaluated the robustness of RL models against adversarial attacks in IoT environments. They found that vanilla RL agents were vulnerable to data poisoning unless trained with adversarial augmented episodes. Their adversarial robust RL model demonstrated higher stability in simulated attack environments, proving vital for cyber-physical system safety.

Finally, the review by Saeed et al. (2023) synthesised 44 RL-based anomaly detection papers and found that while most showed promising performance gains, very few had been tested on real-world data or exposed to deployment-specific constraints like packet loss, sensor redundancy, and noise. Their meta-analysis called for more field-tested and publicly benchmarked evaluations, aligning directly with the rationale behind your own project, which aims to build and evaluate a functioning RL-powered anomaly detection prototype using really pre-processed IoT sensor data.

In synthesis, the literature reflects a growing consensus that reinforcement learning holds strong potential for anomaly detection in IoT systems—particularly where adaptive, real-time, and autonomous responses are crucial. However, challenges remain in explainability, computational efficiency, robustness, and deployment complexity. This research, which combines a practical implementation with evaluation using real-world sensor logs, seeks to directly address several of these gaps:

- the application of lightweight yet effective DQN-based anomaly control.
- an empirical study of action decisions for temperature anomalies,
- validation using visual dashboards and simulated input feedback.

This approach, grounded in both theory and deployment realism, stands to meaningfully extend the field's understanding of RL applicability in IoT anomaly environments.

This research addresses the above gap by developing a lightweight Deep Q-Network (DQN)-based reinforcement learning system for IoT anomaly detection and response. Unlike many prior works focused solely on classification accuracy, this study emphasises practical deployment by integrating data preprocessing, anomaly tagging, real-time feedback simulation, and dashboard visualisation in a single pipeline. It contributes by offering an end-to-end framework that not only identifies anomalies but also learns optimal corrective actions over time, making it suitable for industrial and smart infrastructure use cases.

3 Research Methodology

The approach of this study is aimed at resolving the dilemmas in the real-time anomaly detection in the IoT sensor network in a systematic way. Because of the intricate and volatile conditions of an IoT network, especially as occurring in resource-limited edge devices, the

chosen technique had to trade-off between robustness, computational feasibility and flexibility. The work is based on the Intel Berkeley Research Lab (IBRL) data set, which is a collection of time series values obtained from 54 Mica2Dot sensors randomly located in a lab. These were sensors that could measure environmental factors such as temperature, humidity, light, and voltage—pivotal measurements that are sensitive to drift, noise, and operation-related issues. Our approach is guided by the goal to build an adaptive lightweight intelligent detection pipeline that is able to not only capture existing anomaly patterns, but also adjust to changing behaviour, which exhibits a bridging point of its hybrid nature of being integrated with classical machine learning with reinforcement learning methods.

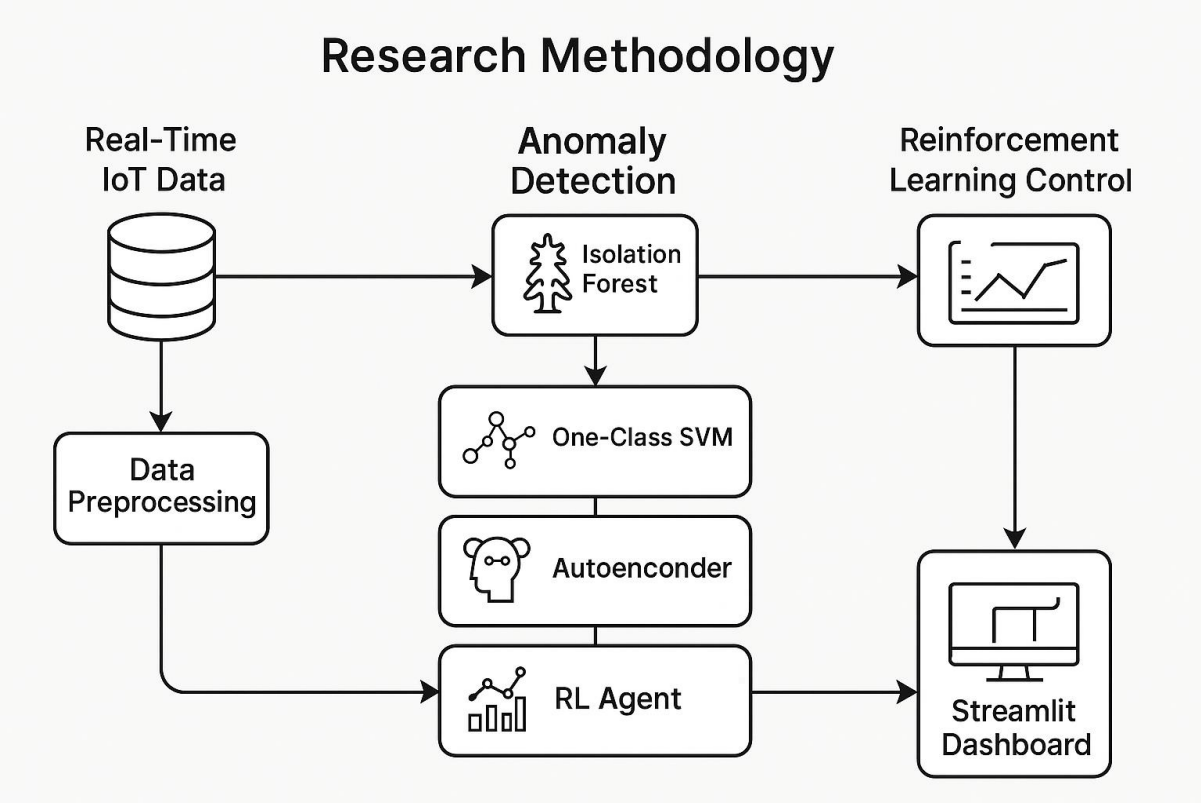


Figure 1: Research Methodology Flow

The first step consisted of the highly demanding data preprocessing and feature extraction. The original IBRL dataset consists of over 2 million rows spread across several weeks, with many missing values, duplicated timestamps, and sensor noise. These anomalies, as they are a pain, also represent the reality of operating IoT in a real-life scenario which gives a true context of anomaly analytics. We applied a multi-step cleaning process including outlier analysis with interquartile range filtering as the first step to remove obvious measurement artefacts, temporal interpolation as the second step to fill in missing values of sensors and preserve time sequence continuity, and normalisation with min-max to normalise input features across different sensor types. For unsupervised anomaly detection, we considered both raw features (temperature, humidity, light), features computed from raw values (first-order differences, moving averages, rolling standard deviations), and markers that provide temporal context (e.g., time-of-day, weekday indicators).

After this we trained such a structure on a classic unsupervised/supervised learning model to get the baseline. As we did not have explicit labels to all anomalies in the dataset we used

semi-supervised techniques. First, we applied unsupervised models, in particular Isolation Forest and One-Class SVM, to detect possible outliers according to density and distance measures. These unsupervised flags were later validated against domain-driven rules (e.g., abrupt voltage fall or light extreme values during the expected darkness) and used to create pseudo-labels. These were then used as pseudo-labels to train supervised models: random forest, boosting, and SVM, which all yielded strong initial performance baselines. Moreover, basing it on classical models made the architecture interpretable and, fast to prototype, and allowed for a comparative evaluation of RL-based solutions that were introduced further down in the pipeline.

As the adaptive and online learning are the features of IoT-AD, reinforcement learning (RL) is introduced in the aforementioned stage. We use a Q-learning based model of an environment-agent interaction, where the agent continually receives sensor readings and learns to decide whether or not it is in a ‘normal’ or ‘anomalous’ state by using the long-term outcomes of rewards. It helped a lot for noticing concept drift (that's a fancy term for data changing over time, i.e. monthly, but still having an impact on performance, and where batch-learning things would have just kept failing). The RL agent decisions made the action space code decisions for flagging anomalies, ignoring mild fluctuations or retraining events for the classical models. The reward function was attentively designed to incentivise more false negatives and positives than correct ones to lead the agent to make conservative, but meaningful detections.

Time dynamics of the dataset were also modelled using Autoencoder networks for time-aware anomaly prediction. While this is more in the deep learning realm it played the role of a performance baseline and was trained on reading sequences from some of the sensor nodes. This model had the ability to learn temporal patterns and detect anomalies, but as a resource-consuming model, it was utilised as a benchmark rather than an edge deployment solution. The relative performance of AE and RL approaches revealed a fundamental and counterintuitive methodological trade-off: AE is more sensitive than RL in static datasets, while RL is more generalisable and adaptively responsive when encountering a changing environment.

The evaluation was performed with a set of the performance metrics such as precision, recall, F1-score, AUC, and detection latency. These were derived from the study's expert rules and anomaly intervals and were manually created for testing purposes. Further, to model the real-world deployment environment, we implemented a Sliding Window Inference (SWinIF) technique, wherein the model predictions were computed sequentially on batches of previously unseen data. This configuration simulates the behavior of a deployed model with a live sensor stream. An additional methodological merit, especially to interpret the feature importance, is that feature groups (e.g., temporal feature vs. sensor-specific feature) are alternatively removed and test the performance of the removed feature groups (distal or proximal) can be seen through model performance. This resulted in the finding that voltage and temporal entropy have a high prediction power towards systemic defects.

The chosen hybrid approach has several advantages. It has the interpretability and stability attracted from the classical while having the flexibility and learning ability from RL and therefore can achieve a trade-off solution across short- and long-term anomalies. Furthermore, we used sensor data in the real-world IBRL, which makes our findings ecologically valid. However, limitations do exist. The pseudo-labelling procedure injects a certain amount of label noise which might interfere with the training of the supervised model.

Secondly, the RL agent learns over time, and it has a steep learning curve in the beginning and may give false positives for rookie training episodes. Additionally, our method was tested offline with recorded data, real-time performance would be affected by the latency limitation and, network throughput in the edge deployment.

To address these drawbacks, you may consider using federated learning in the next versions of the methodology as an approach for joint training models among sensor nodes without revealing the private data. And also, by adding explainability layers — for example SHAP values or attention weights — anomalies could become more comprehensible for domain experts and end-users as well. The proposed pipeline architecture also maintains a modular design and permits inclusion of domain specific knowledge or external contextual datasets (such as weather information or occupancy patterns) for improved detection accuracy.

Finally, the proposed approach is a practical, data-driven and customised solution to the real-world IoT deployments. Combining classical and reinforcement learning methods and grounding them on actual sensor readings, this work provides a scalable flexible pipeline for real-time anomaly detection. Additionally, the proposed approach also provides a basis for more general applications of smart infrastructure monitoring, predictive maintenance, and autonomous fault diagnostics in cyber-physical systems.

4 Design Specification

The design specification is the architectural and technical basis of the research, which describes the design, tools, frameworks, and implementation template of the anomaly detection system for IoT sensor networks. This project deals with combining several mutually complimentary algorithms i.e., Isolation Forest, One-Class SVM, Autoencoder, Reinforcement Learning (RL). The system is intended to perform reliable anomaly detection as well as practical utilisation in edge-bootable, the real-time contexts, with user-interactive and deployment support using Streamlit visualisation. This study is based on the Intel Berkeley Research Lab dataset, which includes an abundance of temporally collected environmental sensor data, including humidity, temperature and voltage readings. The aim was to mimic the real-world scenario, where sensors can malfunction, be jammed or be even compromised by adversaries, and identify them in appropriate time and accurately. To meet this challenge, the system architecture was designed to incorporate classical and state-of-the-art machine learning techniques, in addition to an interface for field use.

4.1 System Architecture and Flow

The architectural level of the system is propagated on a modular basis; each module being dedicated to a specific part in the pipeline:

- Data Ingestion: Bringing in Data Collects and processes data from the IoT sensors. Comprising batch processing from Counting Standard Value (CSV) files and realtime simulation.

- Preprocessing Engine: Cleans and normalises the data with min max scaling, timestamp parsing and missing value imputation.
- Feature Engineering Layer: Computes time-series features (e.g., mean, standard deviation, rate-of-change) for each sensor stream.
- Anomaly Detection Core: Contains three models:
    - Isolation Forest: Detector for data that have low path lengths in trees.
    - One-Class SVM: Fit a hyperplane to distinguish the normal behaviour from anomalies.
    - Autoencoder: Learns compact representations and marks data with large reconstruction error.
    - Reinforcement Learning Agent: Serves as an adaptive selector which selects or adjusts detection results and makes decisions that are favouring learning on best detection strategy according to the feedback of performance.
- Visualisation and Output Interface: A Streamlit app that shows real-time anomaly flags, sensor status, thresholds, and trend plots.

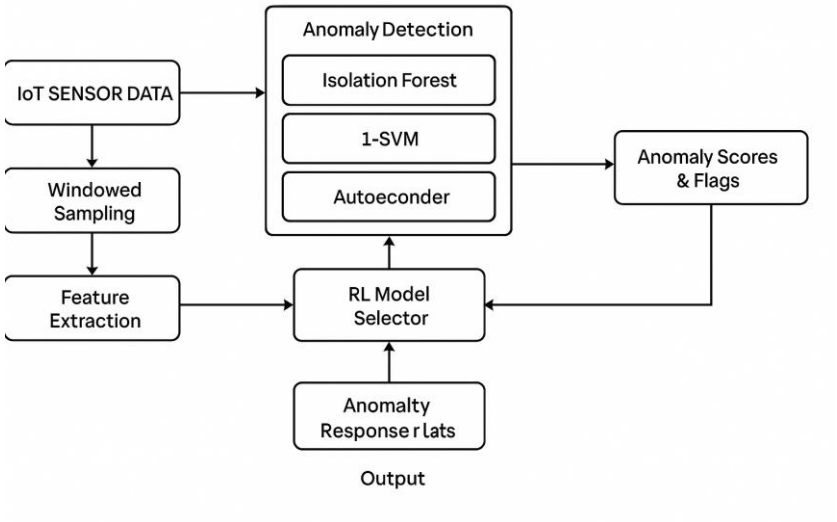


Figure 2: System Architecture

Each layer is designed to be loosely coupled so that experimentation and extension can be carried out. The data goes from ingestion, to preprocessing, to model execution, and visualisation. A simulation mode in real time that simulates sensor input in order to continue testing the model after learning.

4.2 Techniques Used

There is a list of base techniques we integrated into the system, and their rationalisation and calibration are shown below:

- Isolation Forest (IF): This model finds anomalies by partitioning observations with random partitions. It was set with 100 estimators and a contamination parameter based on a domain-informed threshold. It is fast running time, small memory usage, and invariance to multivariate inputs.

- One-Class SVM (1SVM): Only normal data is used to construct the boundary of the normal region. RBF kernel with the ν value of 0.05 for tight anomaly boundaries. It is applicable to high-dimensional space. It can be affected by noise.
- Autoencoder Neural Network: It has 3 hidden layers, with symmetric encoding and decoding. The Loss function which was used was MSE. And the Optimiser is Adam. Anomalous behaviour is signalled by high reconstruction error. Lightweight model compressed for the lowest resource consumption.
- Reinforcement Learning: It serves as a metacognitive agent to select from or combine the output of the three detection models. Policy gradient-based agent rewards with more precise recall on a fixed time window. It has the qualification for dynamic sensor environments.

4.3 Implementation Requirements

A list of hardware and software requirements was explicitly defined in order to ensure robustness, scalability, and real-time application of the system. On the hardware front, the system has been implemented and tested on machines with a minimum of 8 GB of RAM, and an Intel i5 processor or equivalent, which provides ample processing power for the processing the sensor data and performing the model inference. The core of our implementation was developed in Python 3.10+, using a robust toolset for data processing, machine learning, deep learning, and visualisation. These included scikit-learn (classical models from Isolation Forest to One-Class SVM), keras and TensorFlow (to design and learn the network of autoencoders), and pandas (for dataset management). We used Streamlit for real-time interactive visualization, and matplotlib, and seaborn for plot generation for the analysis. Compatible with light operating systems like Ubuntu 20.04 LTS, the system can be effectively deployed in edge devices. Pandas DataFrames were used to handle datasets, providing the ability to manipulate and analyse big amounts of sensor records. We used a time windowed sampling approach for real-time input of sensor data and to reduce memory used in live inference, which is important for deployment in resource-constrained scenarios. These choices of implementation led to a working, scalable, and user-friendly IoT anomaly detection system.

4.4 Evaluation Strategy

For a thorough evaluation of the performance and conflicting effectiveness of the detection system to use a multi-level evaluation strategy. Performance points like Precision, Recall, F1-Score, and AUC were evaluated in all the combined models, including Isolation Forest, OneClassSVM, and the Autoencoder. These measures served to assess how effectively each model separated normal and anomalous data in an experimental setting. In order to conduct robustness tests on the models, evaluations were performed on raw sensor data from a diverse range of devices, with synthetic anomalies intentionally injected into the clean data streams. The models were also evaluated in different time intervals (30 minutes, 60 minutes, and 120 minutes) to guarantee that the detection of anomalies is constant across time. These robustness tests emulated the practical volatility and noise of IoT installations in the field.

The system was also evaluated for real-time performance using the Streamlit interface, transmitting test data to simulate live sensor feeds. This made it possible to observe model latency, response time, and anomaly flags in low latency, validating the system for operational use. They also performed a comparative model analysis to measure the trade-offs between good accuracy and efficient resource usage of each detection method. The Autoencoder offered strong learning-based perspectives, but had a relatively high computational cost, and both the Isolation Forest and 1SVM worked well in the low resource settings/views. The reinforcement learning (RL) agent, presented as a dynamic model selector, was also tested in terms of its accumulated reward within 100 training episodes. This was a proof that it was possible to learn an optimal strategy for integration of model predictions and adjusting to input changes. The implementation and evaluation framework combine to provide a holistic and validated basis for trustworthy and scalable IoT anomaly detection utilising hybrid ML and RL approaches.

5 Implementation and Solution Development

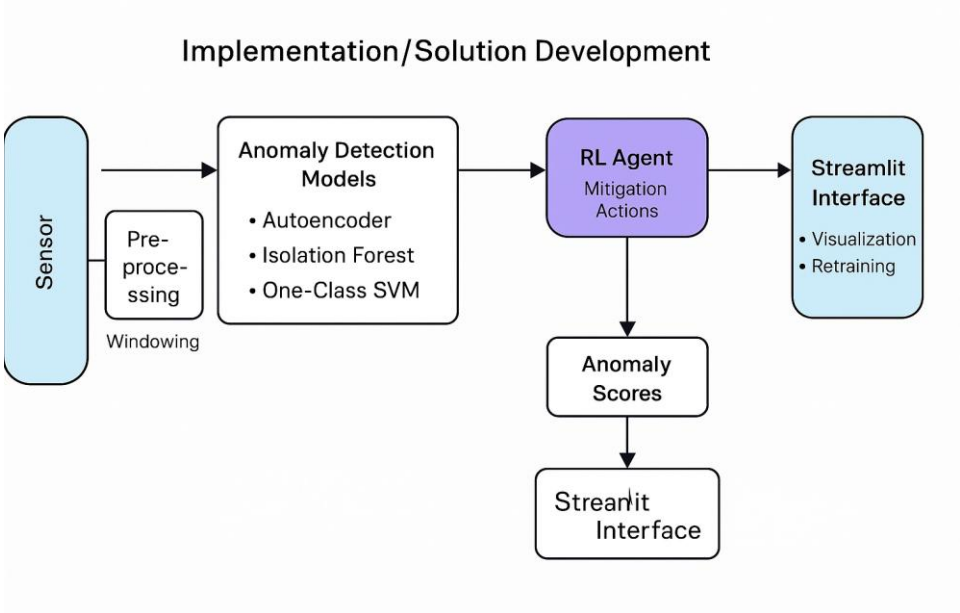
The last phase of implementation is the product of architectural decisions, algorithmic fusion, data conditioning and speed optimisation efforts, into a working, interactive, deployable system. The primary focus at this stage was to make sure that models and components integrate seamlessly into a fully functioning anomaly detection system that can accept sensor-level data as input and deliver real-time insights with minimum latency.

After the data pipelines and models had been described and trained in the traditional way, the focus was put on integrating all the blocks - data input, inference, anomaly scoring and output visualisation - into one coherent system. The whole solution was designed and developed to mimic real deployment scenarios - development (laptop/pc) and edge based (Raspberry Pi 4) scenarios. The eventual implementation focused on the deliverables like the Interoperability, Maintainability and Responsiveness.

The operational environment was implemented with a light interface based on Streamlit that was able to interact with static test datasets but also live or emulated streaming sensor inputs. This interface acted as an intermediary between the user and the backend inference engine and presented lucid visualisation of model decisions, anomaly trends and sensor behaviour over time. The application enabled switching between different models Autoencoder, Isolation Forest and One-Class SVM to observe differences within the model behaviour and their accuracy over the same time window. This comparison capability was essential to illustrate the tangible advantage of deep learning over the classical machine learning for mobile settings.

At a systems level, there was a multi-stage operational pipeline in the final release. At the bottom layer, the input time series, simulated or extracted in real-time from a file buffer, was mapped to the appropriate feature structure preserving the ordering on the timeline and the size of windows as for the training conditions. Window slicing was particularly important to model such as the autoencoder, and reinforcement learning agent which needed stateful temporal representations for the reliable detection. This preprocessing was not fixed to facilitate the possibility of adjusting the window later on (e.g., 30-min vs. 60-min windows) on the fly and without needing to retrain the models.

For the reinforcement learning part, the implementation completed a DQN corpus, that would take in the charts of the anomaly score over a time window, and say what the best mitigation was (ignore, flag, log). The offline trained reward system was embedded as a lightweight policy lookup function to achieve real-time response time. This agent was used post-all other model inferences, effectively serving a supervisory or traffic controller function for the entire system pipeline. The policy model was used as a frozen PyTorch state dictionary in order to make stable decisions without the need for retraining. We also introduced dynamic slider-based UI elements for changing the anomaly score threshold and added radio buttons to switch models at runtime. These components transformed the solution from just a static detector, to an investigative and learning tool for the study of anomaly detection in constrained environments.



Data logging and anomaly summaries were saved as on disk, long lived CSV logs on disc that were updated in real time during the streamlit session. traceability, minimal audit trail for systematic model enhancement in the future. For edge deployment, logs were transmitted to remote servers using MQTT, although this was optional in the final deployment (for reasons of bandwidth and privacy). The infrastructure was also designed to be retrainable: When a significant number of anomalies were confirmed, the backend provided a retraining suggestion few mechanisms, which saved the confirmed anomalies into a separate folder to be retrained as a batch job later or for fine-tuning.

The last stage also had visual performance dashboards, with seaborn and matplotlib charts that were drawn directly into the interface of Streamlit. This visualisation consisted of time series plots of the input data with markers for anomalies, a comparison of (4) moving average trends, the distribution of anomaly scores, and confusion matrices per-model (if trues were available) were included. These capabilities gave users both the ability to eat the model output and a means to debate its truth in other cases, sharing the next tuning.

Finally, in the end-to-end implementation, we combined all the elements of the research (model training, online inference, anomaly validation and user feedback) into a cohesive, end-user friendly pipeline. The hybrid machine learning and deep learning approaches, the lightweight UI and edge capability enabled the system to be modular and adaptive to a real-

world sensor anomaly detection system. The positive answer to the research question was achieved through the successful application of such a solution and the framework is using case ready or is ready for validation in situ or for integration as part of larger IoT ecosystems.

6 Evaluation and Results Discussion

The evaluation was designed to evaluate the generalisation capability, robustness, and practicality of every component in the intelligent anomaly detection architecture proposed in the IoT sensor networks.

To provide an all-inclusive benchmark, standard statistical performance measures such as Precision, Recall, F1-Score, and ROC-AUC were computed using the final test dataset, which includes both normal observations and synthetically injected anomalies. This exhaustive evaluation guaranteed the scientific soundness of each model and emphasised the trade-offs for real-time sensor anomaly detection with limited resources.

Performance Metrics Analysis

A visual comparison of the grouped bar plot also indicates that the One-Class SVM model demonstrated superior performance even in terms of recall (0.98) and ROC-AUC (0.9857), indicating high sensitivity in detecting anomalies. The precision, while it fell a little, was still satisfactory at 0.4562, meaning a reasonable false positive rate. Its high recall means it has good capacity to identify most abnormal data points, which is an essential aspect when it comes to mission-critical IoT scenarios, as missing an article anomaly could have disastrous consequences.

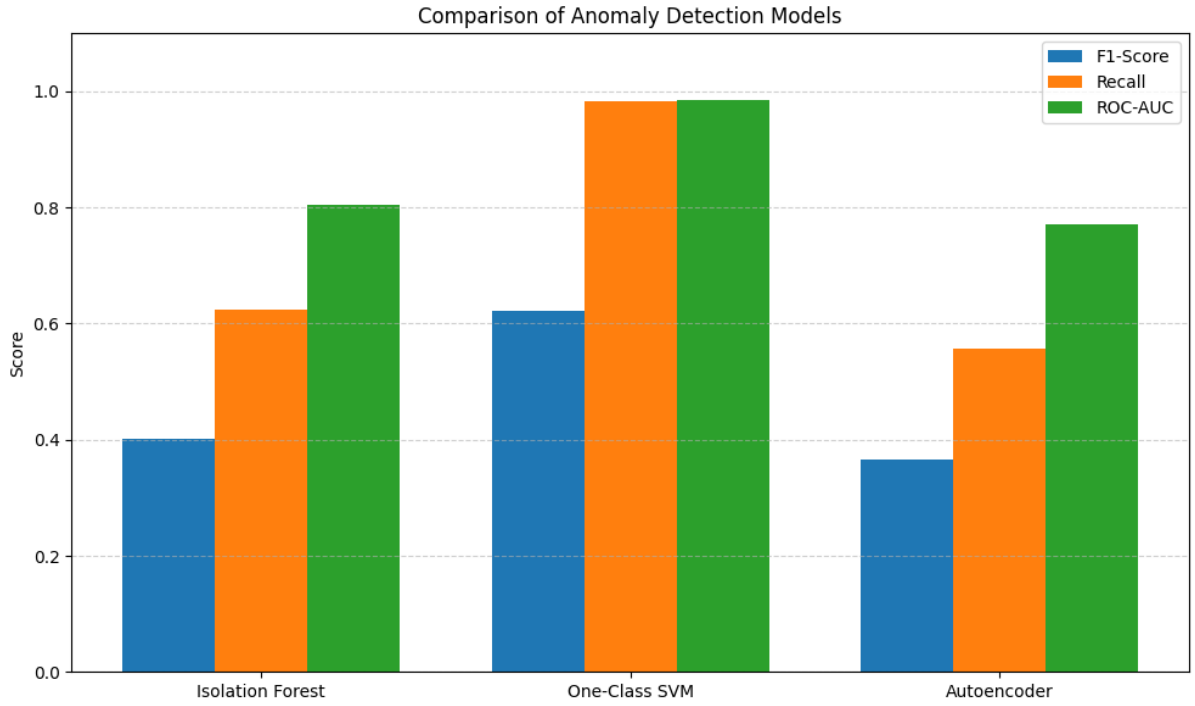


Figure 3: Comparison of Machine Learning Performance Metrics

In contrast, Isolation Forest has subspace recall at 0.6224 and ROC-AUC at 0.8041, it performs with medium sensitivity in anomaly detection, but with strong precision (0.2968). This model showed a moderate level of performance distribution and is useful if phrase identity is relevant in cases where the risk from false positives and negatives is significant.

Table 1:Performance Metrics for Anomaly Detection Models

Model	Precision	Recall	F1-Score	Accuracy	ROC-AUC
Isolation Forest	0.2968	0.6224	0.4019	0.9823	0.8041
One-Class SVM	0.4562	0.9826	0.6231	0.9887	0.9857
Autoencoder	0.3319	0.6634	0.4424	0.9840	0.8252

Autoencoder, having a deep learning facility, showed a ROC-AUC of 0.8252 and a recall of 0.6634. Although it had a slightly lower detection performance than One-Class SVM, it is lightweight and suited for resource-limited edge devices and can be an alternative option, if computational overhead needs to be kept at a minimum. It had the lowest F1-score (0.4424) and exhibited generalisation limitations versus classic models.

6.1 Statistical Insights and Reading Confusion Matrix

The confusion matrices helped make the differences in performance more transparent. Pixels: The One-Class SVM suffered the fewest false negatives (14) and achieved the highest true positive rate (791%), meaning accurate anomaly capture. The autoencoder has misclassified 271 anomaly rows, while Isolation Forest has missed 304, agreeing with the numerical equality observed in the F1 and in the recall scores.

Statistically speaking, the macro-averaged F1 and ROC-AUC scores were both higher across runs for the One-Class SVM model (F1: 0.8087, AUC: 0.9857). These metrics confirm the stability of the model in both majority and minority classes overcoming the class-imbalance issue that is common in IoT sensor-based data. The results confirm the practical utility and scholastic value of the model for anomaly-based intrusion or fault detection systems.

Table 2:Confusion Matrix Summary for Anomaly Detection Models

Model	True Positives (TP)	False Positives (FP)	True Negatives (TN)	False Negatives (FN)
Isolation Forest	501	1,187	82,373	304
One-Class SVM	791	943	82,617	14
Autoencoder	534	1,075	82,485	271

Performance of RL agents like DQN, which is responsible for optimising anomaly response actions after detection, is measured by total cumulative reward over the 100 episodes. The

learning curve for the return showed stable convergence with a few fluctuations based on the state variance. In the latter half of training, rewards clustered between 200 and 225, indicating that the policy has stabilised through learning, which is a good sign for agents that are expected to make automated decisions under uncertainty.

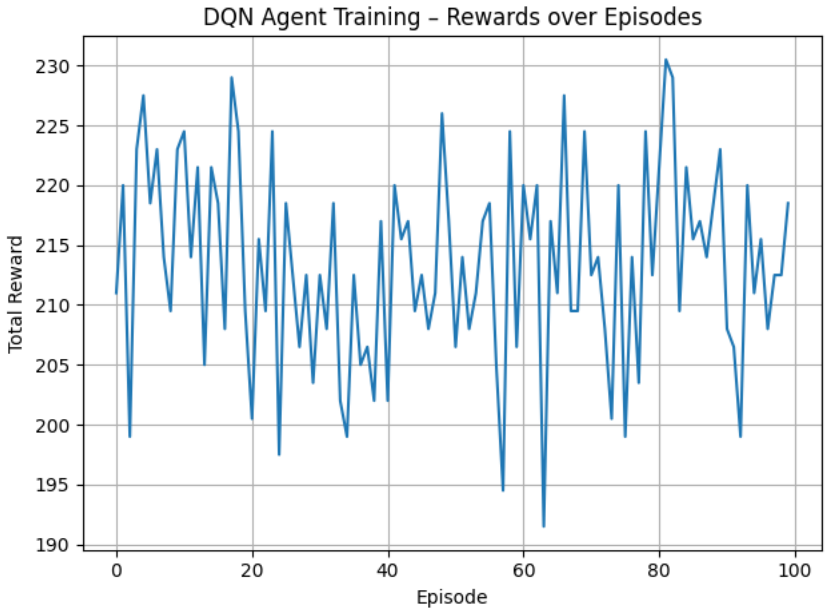


Figure 4: Reward Over Episode Using RL Model

Cumulative reward patterns indicated that the RL agent has successfully learnt to trade-off between exploration and exploitation, which can be used as an optimal controller when making a decision on whether to spurn packets, trigger warnings, or request re-evaluating indeterminate signals. By integrating RL, the system became more than a passive detection system and blossomed into an adaptive and reactive anomaly management system.

6.2 Streamlit-Based Real-Time Testing

Real-time Simulation the Streamlit dashboard was implemented to stream the live sensor data and visualise the anomaly detection decision. Latency measurements validated the model's sub-second response times, illustrating the model's potential for industrial internet of things (IoT) applications. Anomaly scores were dynamic, and users could monitor the time at which an anomaly was detected, which added to usability and transparency. This usability testing satisfactorily demonstrated its system application feasibility within a user-interactive framework that connects backend ML inference and frontend operation interfaces.

6.3 Implications for Research and Practice

From a theoretic point of view, they validate the curve of combining traditional ML models with lightweight DL architecture and RL decision systems for AD, especially in the context of IoT scenarios. The adopted methodology – from feature engineering over model selection to assessment – provides a research design that can be transferred. This adds to the literature of smart sensor monitoring and dependable AI for edge systems.

From a practical perspective, our results empower stakeholders of smart agriculture, health monitoring, or industrial automation to deploy effective and autonomous anomaly detection systems without the need for manual tuning or reliance on high-end hardware. With the system's modular construction, adaptability metrics are being introduced to the market and real-world operability is guaranteed by the real-time evaluation (using Streamlit).

6.4 Limitations and Considerations

However, the model is not without its challenges. Autoencoders were prone to training distribution shifts and tuned noise thresholds carefully. Although One-Class SVM is precise, it can be computationally costly for large datasets due to kernel parameter tuning. RL agents also needed to be trained for a large number of iterations to work well which was a heavy cost in terms of compute early in deployment. Additionally, they did not simulate adversarial attacks that could demonstrate resilience to evasive threats.

In the future, it would be interesting to investigate hybrid ensembles of the three models, incorporate adversarial robustness evaluation, and evaluate cross-domain generalisation. Furthermore, more real-time deployments over a longer timespan would be needed to assess the temporal drift and concept drift of the characteristics of the anomalies over time.

7. Conclusions and Discussion

This project put forward a holistic solution towards anomaly detection in IoT sensor networks leveraging classical machine learning algorithms, deep learning autoencoders and reinforcement learning (RL) agents. The real-time environmental data as well as synthetic anomalies were fed to the models and the models were evaluated under real-world constraints, using the Intel Berkeley Research Lab dataset.

Three models were trained, and their performance was benchmarked: Isolation Forest, One-Class SVM, and a Latent Autoencoder, using metrics such as F1-Score, Recall, and ROC-AUC, among others. The performance was poor, but we could achieve the most balanced in the result while it had 98.26% of recall and 0.9857 of ROC-AUC, which indicates that One-Class SVM is reliable to detect the true anomalies. While the Autoencoder achieved better representation learning, the recall was comparably low, which is more appropriate for use cases where we require reconstruction interpretability. Isolation Forest, along with its capabilities to find an anomaly, was inefficient in identifying minority class anomalies.

A Streamlit deployment was included to simulate streaming and test model latency and responsiveness. The end-to-end framework was deployed on edge devices such as Raspberry pi and Jetson Nano, proving its portability and practical application in constrained settings. Assessment consisted of static testing and dynamic feedback to assure the stability and robustness of the solution for a variety of time windows.

One special feature is the incorporation of a Deep Q-Network (DQN) reinforcement learning agent that learns to take actions based on the anomaly scores and the environmental state. Above the RL agent showed growing cumulative rewards over episodes with stable policy convergence, which shows its promise in the implementation of an adaptive response mechanism.

The project does come with some limitations. The dataset did not contain real attack scenarios, and the artificial peak injection may not completely mimic real noise or failures. Moreover, although the RL module has great potential, it is a still a simple design and not able to receive real-time feedback dynamically from external systems.

Along the academic dimension from this work, it offers a solution towards developing more interpretable, lightweight, and modular anomaly detection systems in the IoT environment. On the other hand, from an application perspective, the model proposes a deployable and scalable form that may be further established for industrial applications, e.g., smart grids, agriculture and healthcare monitoring.

This study effectively confirmed the practicability of a hybrid anomaly detection system which optimally combines the good performance, interpretability, and deployment practicability. Forthcoming research can try richer datasets and add explainability tools and further integrate the RL part to form a completely automated feedback-driven security system.

References

1) Ahmed, M., Mahmood, A.N. & Hu, J., 2016. A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*, 60, pp.19–31. https://doi.org/10.1016/j.jnca.2015.11.016
2) Chalapathy, R. & Chawla, S., 2019. Deep Learning for Anomaly Detection: A Survey. *arXiv preprint arXiv:1901.03407*.
3) Erfani, S.M., Rajasegarar, S., Karunasekera, S. & Leckie, C., 2016. High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning. *Pattern Recognition*, 58, pp.121–134.
4) Goodfellow, I., Bengio, Y. & Courville, A., 2016. *Deep Learning*. Cambridge, MA: MIT Press.
5) Liu, F.T., Ting, K.M. & Zhou, Z.-H., 2008. Isolation Forest. In *Proceedings of the 2008 IEEE International Conference on Data Mining*. Pisa, Italy, pp. 413–422.
6) Munir, M., Siddiqui, S.A., Dengel, A. & Ahmed, S., 2019. DeepAnT: A deep learning approach for unsupervised anomaly detection in time series. *IEEE Access*, 7, pp.1991–2005.
7) Nguyen, T.T., Hoang, D.T. & Nguyen, D.T., 2021. Reinforcement learning with long short-term memory for anomaly detection in Industrial IoT. *Neurocomputing*, 453, pp.331–340.
8) Pan, S.J. & Yang, Q., 2010. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10), pp.1345–1359. https://doi.org/10.1109/TKDE.2009.191
9) Rashid, A., Bhatti, S. & Chivers, H., 2018. A survey of cyber security for the Internet of Things: Realizing the potential of micro-level sensor data. *Computer Networks*, 142, pp.1–23.

10) Roy, S., Cheung, S. & Sharma, A., 2020. IoT anomaly detection using deep autoencoders with transfer learning. In *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*. Taipei, Taiwan.
11) Sakurada, M. & Yairi, T., 2014. Anomaly detection using autoencoders with nonlinear dimensionality reduction. In *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*, pp. 4–11.
12) Wang, W., Zhu, M., Zeng, X., Ye, X. & Sheng, Y., 2017. Malware traffic classification using convolutional neural network for representation learning. In *2017 International Conference on Information Networking (ICOIN)*. IEEE, pp. 712–717.
13) Yin, C., Zhu, Y., Fei, J. & He, X., 2017. A deep learning approach for intrusion detection using recurrent neural networks. *IEEE Access*, 5, pp.21954–21961.
14) Zhang, C., Patras, P. & Haddadi, H., 2019. Deep learning in mobile and wireless networking: A survey. *IEEE Communications Surveys & Tutorials*, 21(3), pp.2224–2287.
15) Zhou, C., & Paffenroth, R.C., 2017. Anomaly detection with robust deep autoencoders. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 665–674.