Edge-AI Predictive Monitoring for Vaccine Refrigeration in Rural Ghanaian Health Clinics

A Combined Edge AI, Embedded Systems, and IoT Approach to Cold-Chain Integrity and Predictive Fault Detection

Adu-Gyamfi Kwadwo

B.Sc. Electrical/Electronic Engineering, Kwame Nkrumah University of Science and Technology

Version 2, November 2026

Preprint, not peer reviewed. This is version 2 of EngrXiv preprint 10.31224/8149 (https://doi.org/10.31224/8149).

Table of Contents

Abstract 3
1. Introduction 4
2. Related Work 5
3. Proposed System 6
4. Methodology 7
5. System Implementation 11
6. Results 14
7. Discussion 21
8. Future Work 23
9. Conclusion 24
Data and Code Availability 25
Declarations 26
References 27

Abstract

Vaccine cold-chain failures remain a major and preventable cause of vaccine wastage worldwide, and the problem is especially acute in rural, resource-constrained health facilities where unreliable electricity, limited technical support, and unreliable internet connectivity compound the risk. This work presents the design, testing, and prototype implementation of an edge-AI-informed monitoring system for vaccine refrigeration units, motivated by the documented cold-chain challenges faced by rural health clinics in Ghana. Existing systems are largely reactive, alerting only after a temperature threshold has been breached, and many depend on the cloud. The proposed system combines three detection layers that run locally on a low-cost ESP32-S3 microcontroller: (1) predictive compressor-fault detection based on a machine-learning analysis of cycling behaviour, (2) gas-based detection of likely non-vaccine items placed in the unit, and (3) cumulative exposure tracking using a real-time clock. The predictive approach was first tried on the public MetroPT-3 dataset of a failing metro-train air compressor. With a time-ordered train and test split, it flagged 81 of 83 documented failure hours and 4.7% of later normal hours (ROC AUC 0.98), although the result depends on how a compressor cycle is defined. It was then applied to a purpose-built simulation of vaccine-fridge degradation. The analysis was turned into lightweight threshold rules that run on the microcontroller and were tested in a simulated hardware environment (Wokwi), including an MQTT reporting layer and a web dashboard. Decision-logic timing and the effect of network conditions on loop timing were also measured on a real ESP32-S3 using synthetic inputs. The rule was also replayed on recorded power traces of household fridges and freezers from three UK homes (REFIT dataset). With the initial settings it raised false Warnings between about 5 and 28 times per 30 days on healthy appliances. A longer calibration with a persistence requirement, chosen by leave-one-house-out testing, reduced false Warnings by about a third to 70% and false Criticals by about two-thirds or more, at the cost of slower detection of injected faults, and false alarms did not disappear. These settings are now the firmware default, and a replay of the recorded phases on an ESP32-S3 gave the same alert sequences as the analysis. An Isolation Forest on window features did not do better than the ratio rule on the same traces. The complete firmware was also run on the board with temperature, door and gas values generated by the sketch, and it went through calibration, Warning, Critical, recovery, the gas alert and the exposure alert without a reset. The rule was finally run on the labelled records of ten real ultra-low-temperature laboratory freezers (about 89 freezer-years, 22 service-dated faults). It raised about 0.15 Warning episodes per 30 days in healthy time, but it flagged only 4 of 14 compressor or refrigerant faults in the 90 days before their service dates, and in 12 of 18 long unlabelled episodes the freezer's own warm alarm, shutdown or failure to reach its setpoint followed within 10 days. The complete system has not yet been tested with physical sensors or on a physical refrigerator. This document describes the related work, the methods, the system design, the results, and the limitations and next steps.

Keywords: vaccine cold chain, predictive maintenance, edge AI, ESP32-S3, anomaly detection, Isolation Forest, IoT, Ghana.

> ***Note on terminology.*** *This work uses the term "edge AI" to describe a system whose decision-making runs locally on the embedded device rather than in the cloud, and whose logic is derived from machine-learning analysis of data. It does not claim that a trained machine-learning model runs its inference directly on the microcontroller. As explained in Section 4.3, the Isolation Forest model used in this work runs offline, on a workstation, to find a predictive signal in the data; that signal is then turned into lightweight, explainable threshold rules that run on the ESP32-S3 in real time. This is a common engineering pattern for constrained edge hardware, and it is stated here so that the scope of the term "edge AI" is clear throughout this document.*

1. Introduction

1.1 Background and Problem Statement

The World Health Organization estimates that a substantial proportion of vaccines produced worldwide are wasted every year, in large part due to cold-chain failures, which are temperature excursions during storage or transport that compromise vaccine potency. Beyond the direct loss of vaccine stock, an undetected refrigeration failure also imposes a secondary cost: unplanned equipment downtime, emergency repair costs, and disruption to a facility's ability to serve patients while the unit is out of service.

The overwhelming majority of existing vaccine cold-chain monitoring solutions are reactive rather than predictive: they measure temperature and raise an alert only once a threshold has already been crossed, by which point the vaccines inside may already be compromised. Very few systems attempt to anticipate an impending equipment failure before it manifests as a temperature excursion, and fewer still are designed to remain fully functional when internet connectivity is unreliable or absent, which is common in the settings where this problem is most severe.

1.2 Motivation: The Ghana Context

This work is grounded in the documented, real cold-chain challenges faced by health facilities in Ghana. During the national COVID-19 vaccination effort, Ghana's rollout was significantly constrained by frequent power outages and a shortage of temperature-controlled transport, at one point, only fourteen temperature-controlled vehicles existed in the entire country. Reporting from UNICEF describing a rural health centre in the Dokrochiwa community illustrates the human dimension of this problem directly: an unreliable refrigeration unit forced staff to hold only two weeks of vaccine stock at a time rather than a full month, causing mothers to travel long distances only to find that stock had run out, eroding trust in the health system.

Ghana Health Service assessments attribute these failures principally to unreliable electricity supply, aging equipment, and the particular vulnerability of rural CHPS (Community-based Health Planning and Services) compounds. In response, UNICEF and Gavi have supplied thousands of pieces of cold-chain equipment to Ghana, including more than two hundred solar-powered refrigeration units. This equipment, however, still lacks any predictive monitoring layer: a solar-powered unit in a remote clinic can fail quietly, with the failure discovered only when a staff member opens the door and finds the contents already warm.

The gap this work addresses is reliable equipment without predictive monitoring, deployed where power and connectivity are unreliable.

1.3 Research Objectives

- To review existing vaccine cold-chain monitoring and industrial predictive-maintenance literature and identify a concrete, addressable gap.
- To test a machine-learning-based predictive fault-detection approach against genuine, real-world equipment failure data before applying it to the target domain.
- To apply and adapt this approach to a vaccine-refrigeration-specific context, given the absence of any public dataset of real vaccine-fridge failures.
- To design and implement a low-cost, edge-AI-capable embedded prototype that performs predictive fault detection, misuse detection, and cumulative exposure tracking without dependence on continuous internet connectivity.
- To add an optional IoT reporting layer for remote visibility, explicitly designed as a convenience layer rather than a dependency of the core safety functions.

2. Related Work

Existing approaches to this problem fall into three broad categories, reviewed below, each of which addresses part of the problem but not the combination this work proposes.

2.1 Basic IoT-Based Cold-Chain Monitoring

The most common category of existing system pairs a microcontroller (typically an ESP32) with a temperature and humidity sensor, streaming readings to a cloud dashboard, most commonly Firebase and triggering an SMS or buzzer alert when a fixed temperature threshold is crossed. Representative examples in the literature include an ESP32/DHT22/HX711-based system for underserved-area vaccine storage with SMS alerting and a chatbot interface; an ESP32/SHT31-based design that stores readings to both Firebase and local SD-card storage for resilience against connectivity loss; an open-source ESP32 and GPS-based system for tracking vaccines in transit; and an Arduino-based supply-chain monitoring system incorporating location and quantity tracking. These systems are valuable and widely deployed, but they share a common limitation: they are reactive by design, only raising an alert once a threshold has already been breached, and most depend on continuous cloud connectivity to function at all.

2.2 AI-Based Anomaly Detection for Vaccine Refrigeration

A smaller body of work applies machine learning directly to vaccine-fridge temperature monitoring. A semi-supervised convolutional-autoencoder approach has been demonstrated running directly on an ESP32-class microcontroller to detect subtle temperature anomalies in vaccine refrigeration that a simple fixed threshold would miss, reporting approximately 92% detection accuracy in testing. This represents a meaningful advance over threshold-based alerting, since it can catch abnormal patterns before they cross a hard limit. However, this class of system monitors the vaccine's thermal environment; it does not monitor the health of the refrigeration equipment itself, and therefore cannot provide advance warning of an impending compressor failure before the temperature is already affected.

2.3 TinyML Predictive Maintenance in Industrial Refrigeration

A separate and more mature body of work applies TinyML techniques to predictive maintenance of refrigeration and cooling equipment in general industrial settings, unrelated to vaccine storage specifically. Documented examples include vibration-based fault detection running on ESP32 hardware, a lightweight edge-AI system on the ESP32-S3 combining on-device inference with fault-type and time-to-failure estimation, and current-based compressor failure detection using an Isolation Forest model. These approaches demonstrate that predictive, on-device fault detection for compressor-driven cooling equipment is both feasible and effective. However, none of this work has been adapted to, or validated within, the specific context of hospital or clinic vaccine refrigeration.

2.4 Identified Research Gap

No existing system in the reviewed literature combines all three of the following properties in a single, low-cost device: (1) predictive detection of an impending refrigeration-equipment failure, ahead of any temperature excursion; (2) tracking of cumulative vaccine-exposure risk derived from the full temperature history, rather than a single threshold breach; and (3) fully local, on-device operation that continues to function correctly when internet connectivity is unreliable or unavailable, which is the usual situation in the rural clinic settings where this problem is most severe. This is the gap this work addresses.

3. Proposed System

3.1 System Overview

The proposed system is built around a single ESP32-S3 microcontroller and combines three independent detection layers with a local alerting mechanism and an optional IoT reporting layer. Table 1 summarises the system's functional layers.

Layer	Function	Detection Basis
Predictive fault detection	Warns of likely compressor degradation before a temperature excursion occurs	Machine-learning-derived compressor on/off cycling signature, turned into on-device threshold rules
Misuse / gas detection	Flags likely non-vaccine items (e.g. alcohol-based drinks) placed in the unit	MQ2 combustible-gas/vapour sensor, rule-based threshold
Cumulative exposure tracking	Tracks total time-above-threshold exposure per day, not just instantaneous breaches	DS1307 real-time clock, cumulative timer logic
Local alerting	Immediate, distinguishable audible and visual alerts per condition, independent of connectivity	Status LEDs (green/yellow/red/compressor) and a multi-tone buzzer
IoT reporting (optional)	Remote live visibility of device status via a web dashboard	Wi-Fi, MQTT publish/subscribe, Streamlit dashboard

3.2 Design Rationale

Two design decisions shaped the system. First, the predictive and misuse-detection layers are designed to operate entirely locally: the device's core safety behaviour (sensing, fault detection and alerting) does not depend on Wi-Fi or MQTT connectivity being available or successful. This directly reflects the connectivity constraints documented in the Ghana context described in Section 1.2. Second, rather than deploying the trained machine-learning model itself onto the microcontroller, the system uses the model to discover a predictive signal from data, which is then turned into lightweight, explainable threshold rules suitable for real-time execution on constrained hardware. This decision, and the reasoning behind it, is discussed in detail in Section 4.3.

4. Methodology

4.1 Stage 1: Testing on Real-World Industrial Data

Because no public dataset of real vaccine-refrigerator failures exists, the detection method was first tried on a genuine, independently collected dataset from a different system: the MetroPT-3 dataset, published by INESC TEC and the University of Porto through the UCI Machine Learning Repository. It contains sensor readings from the air compressor of a metro train, including motor current, several pressure channels and oil temperature, from February to August 2020, together with four documented air-leak failure windows. The dataset documentation describes logging at 1 Hz. The file used here contains every tenth sample (1,516,948 rows, a 10-second interval).

An Isolation Forest was trained only on normal-operation hours, using seven features computed over hourly windows: average compressor on-time, average off-time, signal variability, peak and minimum readings, cycle count, and a three-hour rolling average of cycle count. The compressor was counted as ON when the motor current was above 5.5 A. The dataset documentation gives about 4 A for offloaded running and about 7 A under load, so this level separates loaded from offloaded operation. An hour was labelled a failure hour if at least 30 minutes of it fell inside a documented failure window, which gave 83 failure hours and 4,131 normal hours. The model was fitted on the earliest 60% of normal hours and tested on the later normal hours and all failure hours.

On this held-out test the model flagged 81 of 83 failure hours, covering all four events, and 77 of 1,653 later normal hours (4.7%), with a ROC AUC of 0.98 (Section 6.1). Three points limit what this shows. First, the result depends on how a compressor cycle is defined: counting the motor as ON above 1.0 A, or using the outlet-valve signal, flagged only 22 and 24 of the 83 failure hours, and the 5.5 A definition was the best of the three tried. Second, about half of the flags in the test were false, because normal hours outnumber failure hours roughly 20 to 1. Third, flags are per hour and say nothing about how early a failure is detected, and the failure windows come from the dataset documentation and were not independently checked. The loaded and offloaded cycling of an air compressor also has no direct counterpart in a refrigerator, so this stage tests the method on a different machine and is not evidence of vaccine-fridge performance.

4.2 Stage 2: Application to a Domain-Specific Simulation

With the method tried on genuine failure data, the same feature extraction and modelling approach was applied to the target domain. A simple simulation of a vaccine refrigerator was written in Python. It models a 2 to 8 °C thermostat with compressor cycling, random door-open events, and a gradual compressor degradation that starts on day 20 of a 30-day run, implemented as a progressive loss of cooling efficiency and an increase in warming rate. The degradation is deliberately gradual, to reflect how compressor wear develops over time and not as an instant failure.

The same hourly features and Isolation Forest were used. The model was fitted on the earliest 60% of healthy hours (288 hours) and tested on the later 192 healthy hours and the 240 degrading hours. The average anomaly score fell as wear progressed, from about 0.10 in days 20 to 23, to 0.07 in days 23 to 26 and 0.03 in days 26 to 30, against 0.09 for the held-out healthy hours (Figure 1). Individual hours overlap heavily, however: the ROC AUC for separating healthy from degrading hours is only 0.64, so with this feature set the model is a weak hour-by-hour detector. Section 6.2 gives the flag counts.

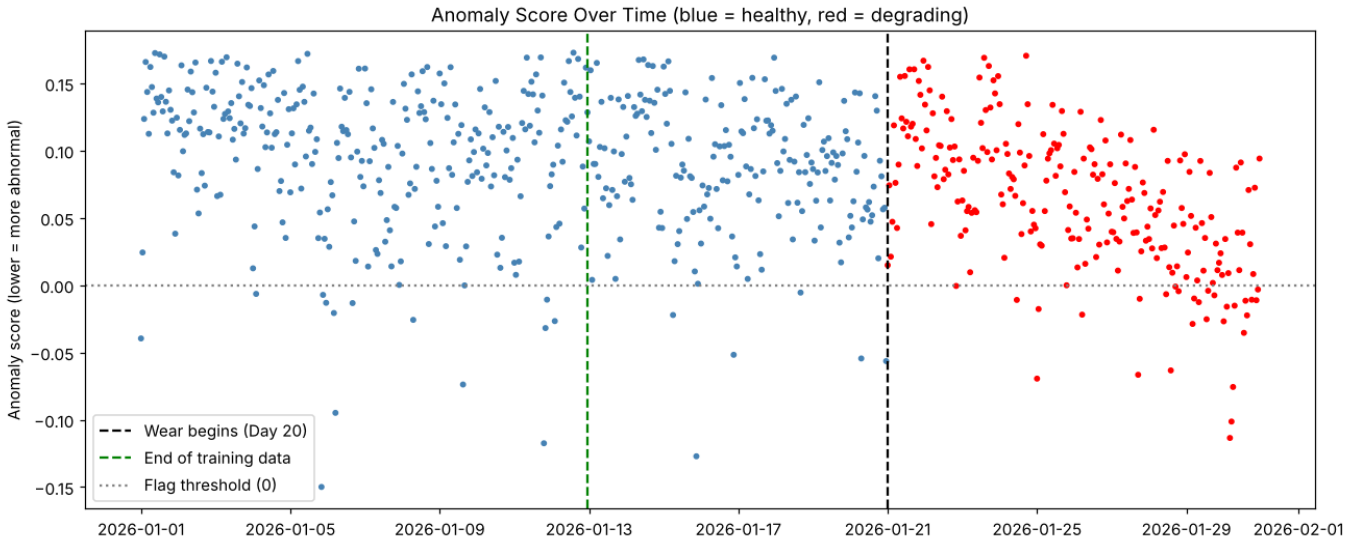

Figure 1. Anomaly score over the simulated run. Blue points are healthy hours and red points are hours after wear begins (day 20). The green line marks the end of the data the model was trained on. The average score declines as wear increases, but individual hours overlap heavily with healthy ones.

4.3 From Offline Analysis to Embedded Rules

The Isolation Forest model itself was deliberately not deployed onto the ESP32-S3. This was a considered engineering decision, not an oversight: Isolation Forest models comprise on the order of a hundred individual decision trees with no standard, low-risk export path to embedded C++, unlike some simpler model classes. Attempting a full, correct export within the available project timeline carried a material risk of introducing subtle numerical errors that would be difficult to detect without extensive on-device testing.

Instead, the model was used offline to look for the underlying signal in the data, and that signal was turned into simple rules that run on the microcontroller. This is not formal model distillation: no smaller model was trained on the Isolation Forest's outputs. The signal is the ratio between compressor on-time and off-time. In the simulation, a healthy unit ran with an average ON/OFF ratio of about 0.52 (roughly 63 simulated minutes on and 120 minutes off), and over the last three days of wear the ratio reached about 1.7, around 3.3 times the healthy value. The on-device logic calibrates a per-unit baseline over an initial period and then compares a rolling average of the five most recent on/off cycles with that baseline. Smoothing is needed because single cycles are noisy, as Section 4.4 shows.

4.4 Threshold Check Against the Simulation

The Warning threshold (1.8 times the baseline ratio) and the Critical threshold (3.5 times) are design choices. They were checked against the simulation, not against a real refrigerator, with the cycle-level analysis in threshold_analysis.py. Door events make single cycles noisy: in the healthy days, one ON/OFF pair exceeded 1.8 times the baseline in 18% of cases (maximum 10 times) and a five-cycle average did so in 2% of windows (maximum 2.0 times). Once door-affected cycles were left out, no window exceeded 1.17 times. This is why the firmware uses both a five-cycle average and a door-event exclusion. With door cycles left out, the Warning threshold gave no warnings in the 20 healthy days, was crossed in 26% of windows in days 23 to 26, and was crossed in all windows in days 26 to 30 (average drift 2.55). The Critical threshold was never reached: the highest drift was 3.04, so 3.5 is a margin for more severe degradation than the simulation produces. A baseline taken from the first three ON and OFF cycles, as the first version of the firmware did, gave similar results.

4.5 Replay on Household Fridge Power Traces

To see how the on/off-cycle rule behaves on real compressor cycling, it was replayed on the REFIT Electrical Load Measurements dataset (cleaned version; Murray, Stankovic and Stankovic, Scientific Data 2017; CC BY 4.0). REFIT records the active power of individual appliances in UK homes at a median interval of about 7 s. Four appliance plugs were used: the fridge-freezers in House 2 and House 15, and the separate fridge and freezer in House 20. They cover 460 to 617 days each. The dataset metadata does not record the models of these appliances. These are domestic units, not vaccine refrigerators, so the test shows how the rule copes with doors, food loading, defrost and seasons. It says nothing about temperature sensing or the other layers of the system.

Compressor phases were extracted from the power trace: ON at 30 W or more and OFF at 10 W or less, with hysteresis. Cycles never cross a gap of more than 2 minutes in the timestamps. Dips shorter than 2 minutes inside a phase were merged into the surrounding phase, phases longer than 6 hours were dropped, and ON phases whose mean power was above twice the median ON power were dropped as defrost. The cleaned files carry an Issues flag that marks rows where the sub-meters add up to more than the household aggregate. In the first version of the analysis, phases containing these rows were dropped, which removed 27 to 32% of the phases and cut the data into hundreds of short fragments. Because the flag concerns the aggregate and not the timing of one plug, these phases were kept in the main analysis, and the version with them dropped is reported as a check. This change was made after seeing the first results. REFIT does not mark door openings, so no cycles were excluded for doors as the first version of the firmware did, which makes the false-alarm counts pessimistic compared with the firmware.

The original firmware rule was applied to the phase sequence as in Section 4.3: a baseline from the first three ON and three OFF phases, a five-phase rolling average, Warning above 1.8 and Critical above 3.5. The baseline was recalibrated after any outage longer than one hour, as a rebooted device would. False alarms are counted as alert episodes per 30 days, because overlapping windows make percentages of windows overstate the number of independent events. To test detection, a fault was injected into the real durations: in 30-day slices calibrated at the start of each slice, ON durations were multiplied and OFF durations divided by the square root of a factor that grew linearly from 1 to the final ON/OFF ratio multiplier over 7 days, starting on day 15. This tests the detector and not a physical failure.

Finally, 48 variations of the rule were compared: calibration over 3, 6, 12 or 24 phases of each kind, a window of 5 or 10 phases, mean or median, and persistence (an alert only after 1, 3 or 6 consecutive windows above the threshold), with the thresholds unchanged. The choice was made by leave-one-house-out testing: for each house, the best variation was chosen using only the other two houses and then scored on the held-out house. A variation was eligible only if a 3x fault reached Critical in at least 80% of slices with a median delay of 7 days or less on every training appliance, and among eligible variations the one with the fewest healthy false alarms (Warning episodes plus three times Critical episodes) was chosen. This rule was written down before the sweep was run.

4.6 Comparing Detectors and the Warning Threshold

Two further questions were asked on the same REFIT traces. The first was whether another detector would beat the ratio rule, in particular an Isolation Forest on window features, since the forest was the model used offline in Section 4.1. Three detectors were compared, all with 24 calibration phases and persistence 6: the ratio rule of Section 4.3; the same ratio computed from exponentially weighted averages of the log durations (weight 0.1); and an Isolation Forest (50 trees, 128 samples per tree) on five window features, namely the log change of ON time, the log change of OFF time, the log drift, and the coefficient of variation of the ON and OFF durations in the window. To compare them fairly, each detector was given one alarm threshold, chosen on the two training houses so that the healthy false-alarm rate there was at or below a target of 2 or 5 episodes per 30 days, and then applied unchanged to the held-out house. The forest was fitted on the training houses only. Its scores on those houses are in-sample, so its threshold choice is slightly optimistic, while the held-out numbers are not.

The second question was how the Warning threshold of 1.8 trades false alarms against fault size. With 24 calibration phases and persistence 6, the threshold was varied from 1.3 to 2.6, and for each appliance the healthy

Warning episodes per 30 days and the share and delay of detected 1.5x, 2x and 3x injected faults were recorded. This curve is descriptive over the same three homes and is not a held-out test.

Finally, the Isolation Forest was exported for the board. A forest (50 trees, 128 samples per tree, the five features above) was written as constant arrays (feature index, threshold, left and right child, and for each leaf its depth plus the average path length of its training samples), and its alarm threshold was set at the 99.5th percentile of the training scores. The board scored 3,000 stored test vectors (2,000 healthy windows and 1,000 windows from injected 3x faults) as float32 values, and each score was compared with the one scikit-learn gave on a computer. This is a device test of the export, not an evaluation of detection quality.

5. System Implementation

5.1 Hardware Architecture

The prototype was implemented and verified using the Wokwi hardware simulator, targeting an ESP32-S3 development board. Table 2 lists the sensing and actuation components and their function.

Component	Interface	Function
DHT22	Single-wire digital (pull-up resistor)	Temperature and humidity sensing
Push-button	Digital input, internal pull-up	Door-open/close sensing
MQ2 gas sensor	Analogue input	Combustible-vapour / misuse detection
DS1307 real-time clock	I2C	Persistent real-time clock for cumulative exposure tracking
Status LEDs (green / yellow / red)	Digital output	Fridge-health status indication
Status LED (blue)	Digital output	Compressor-running indication
Piezo buzzer	PWM (LEDC hardware tone generation)	Distinguishable audible alerts per condition

5.2 Predictive Fault Detection Module

During an initial calibration period, the device times the first completed ON phases and OFF phases to set a per-unit baseline (24 of each in the current default; the first version used three, and a compile-time switch restores that for quick demonstrations). The first transition after power-up is skipped, because the phase that was running at start-up has an unknown length. Calibration phases also enter the rolling window of the five most recent phases, so the drift is evaluated from the first phase after calibration. An alert is raised only after six consecutive evaluations above the threshold (one in the demonstration setting). When the window holds at least two ON and two OFF durations, a smoothed on/off ratio is computed and compared with the calibrated baseline, as described in Section 4.4. Cycles immediately after a door-open event are excluded from this evaluation, to avoid mistaking door-driven temperature disturbance for compressor degradation.

5.3 Gas / Misuse Detection Module

An MQ2 combustible-gas sensor is polled periodically and compared against an empirically determined threshold, calibrated from observed resting and elevated readings within the simulation environment. It is important to state precisely what this module does and does not detect: the MQ2 sensor responds to combustible gases and vapours (for example alcohol-based drinks) and is not a general food-detection sensor. This scope was chosen because it addresses a real, documented category of vaccine-fridge misuse (storage of food or drink, which international guidance explicitly discourages) within the constraints of the sensor hardware available, without overstating what the sensor can do.

5.4 Cumulative Vaccine Exposure Tracking

A DS1307 real-time clock provides a persistent, battery-backed time source. Whenever the sensed temperature is at or above the safe upper limit, elapsed time is accumulated into a daily counter; if this cumulative warm-time exceeds a configured threshold within a calendar day, a distinct exposure-risk alert is raised, independent of whether any single instantaneous reading looked severe. This directly distinguishes the system from simple threshold-alarm designs,

which cannot detect the cumulative effect of many short excursions. The daily-reset design is a deliberate first-phase simplification, discussed further in Section 7.2.

5.5 IoT Connectivity Layer

The device connects to Wi-Fi and publishes its status (temperature, compressor state, alert level, gas-sensor reading, cumulative exposure, door state and device time) as a JSON payload over MQTT at regular intervals. A companion Python and Streamlit application subscribes to the same topic and shows a live web dashboard (Figure 3), including a notice when no message has arrived for 30 seconds. The Wi-Fi and MQTT code runs in its own task on a separate processor core and reads a status snapshot that the main loop updates, so a slow or unreachable broker cannot hold up sensing and alerting. This matters: in a test on a real ESP32-S3, an MQTT connect call made from the main loop stalled it for up to 17 seconds (Section 6.4). The IoT layer is meant as a convenience and not as a dependency, in line with the connectivity constraints described in Section 1.2.

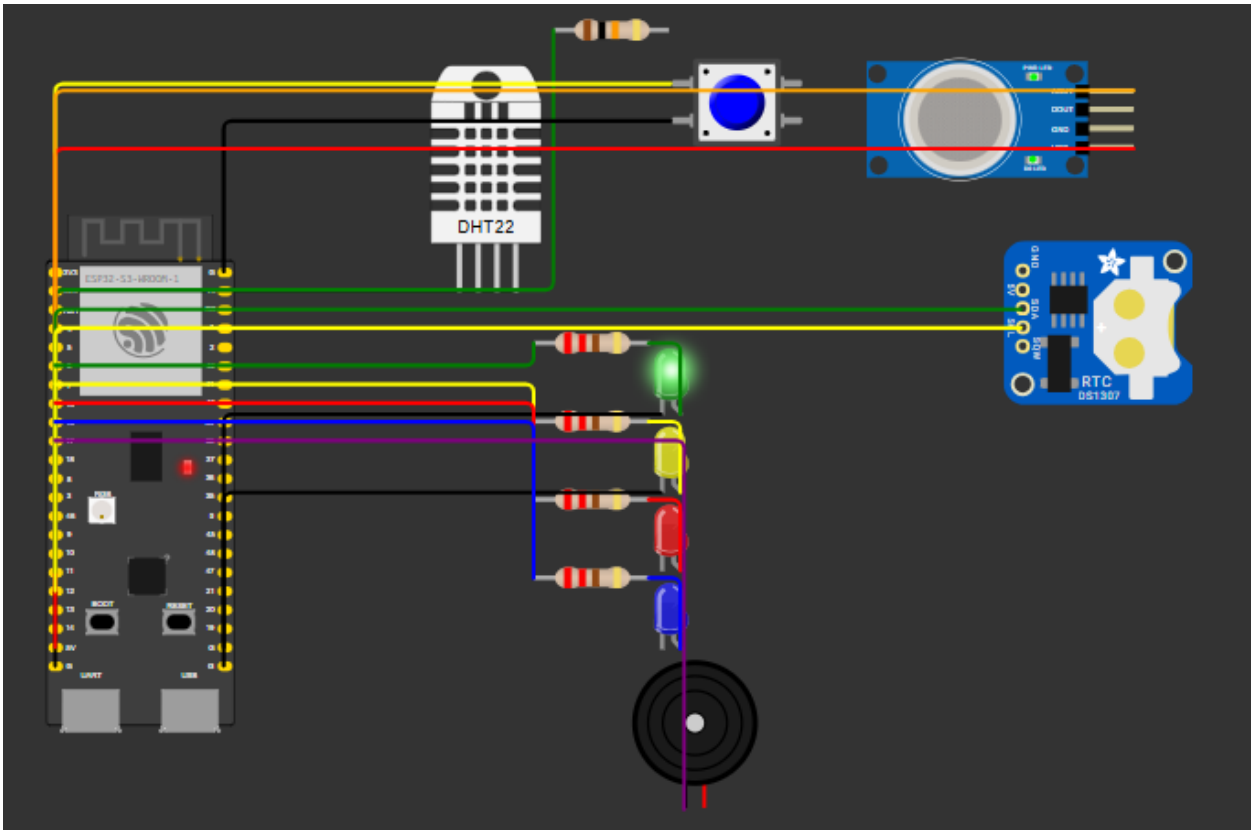

Figure 2. Full prototype circuit (ESP32-S3, DHT22, door button, MQ2 gas sensor, DS1307 RTC, status LEDs, and buzzer)

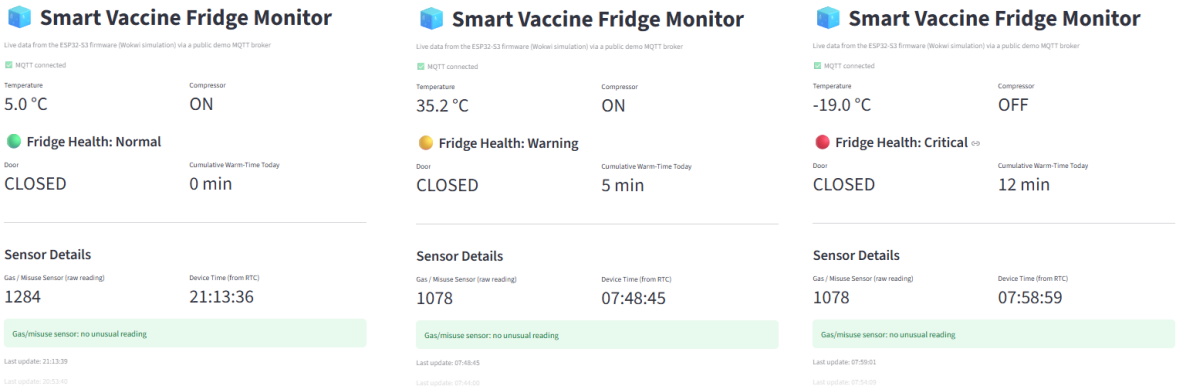

Figure 3. Dashboard in three states (Normal, Warning, Critical), fed by the firmware running in the Wokwi simulator over a public MQTT broker. The temperature was set by hand in the simulator. These are screenshots of test states, not readings from a real refrigerator.

6. Results

6.1 Real-Data Validation Results

Metric	Motor current above 1.0 A	Motor current above 5.5 A	Outlet-valve signal active
Failure hours flagged	22 / 83	81 / 83	24 / 83
Later normal hours flagged	107 / 1,653 (6.5%)	77 / 1,653 (4.7%)	105 / 1,653 (6.4%)
ROC AUC	0.935	0.977	0.937
Average precision	0.26	0.70	0.26

Columns show the three definitions of compressor ON that were tried. Each row of results is from the held-out test: the model was fitted on the earliest 60% of normal hours (to 10 June 2020) and tested on the later normal hours and all failure hours, with contamination 0.05. With the 5.5 A definition, all four failure events were flagged (23 of 24, 6 of 7, 47 of 47 and 5 of 5 failure hours). About half of the flags in that test were false (81 failure hours against 77 normal hours). The failure windows are taken from the dataset documentation, and the third window spans about 53 hours, of which 47 remain after a coverage filter.

6.2 Simulated Fridge Validation Results

Configuration	Later healthy hours flagged	Degrading hours flagged	ROC AUC
Strict (contamination = 0.05)	6 / 192	30 / 240	0.638
Sensitive (contamination = 0.15)	33 / 192	88 / 240	0.638

The model was fitted on the earliest 60% of healthy hours and tested on the later healthy hours and all degrading hours. Of the 168 hours with wear of 0.3 or more, the strict model flagged 29 and the sensitive model 81. The separation between healthy and degrading hours is weak with this feature set (ROC AUC 0.64), although the average anomaly score declines as wear increases (Figure 1). The on-device rule uses the on/off cycle ratio and not this model, and the threshold check in Section 4.4 shows that the ratio separates healthy from degrading cycles much more clearly.

6.3 Embedded System Verification

The original version of the firmware was run in the Wokwi hardware simulator with the temperature driven by hand through the DHT22 slider. Checked behaviours include: DHT22 temperature sensing (after an initial pull-up and read-interval issue was resolved); compressor on/off cycling logic; calibration after three ON and three OFF cycles (baseline ON 38.44 s and OFF 44.91 s in one run); rolling-window smoothing and drift calculation; door-event exclusion; distinguishable audible alerts for warning, critical, misuse and exposure-risk conditions, each repeating until the condition clears; MQ2 gas-sensor threshold behaviour at resting (false) and elevated (true) readings; RTC-based cumulative exposure accumulation and daily reset; and MQTT delivery to the live dashboard (Figure 3). With ON periods stretched relative to OFF, the smoothed drift rose from about 1.5 to 4.3, and the sketch printed Warning at 1.82 and Critical at 4.07. It stepped back down to Warning when the drift fell to 3.34. In later runs the published messages carried alert level 1 (Warning) and alert level 2 (Critical, drift 3.65 at escalation). During recovery the drift fell through 2.48, 2.58, 2.22, 2.29, 2.01, 2.04 and 1.82 (still Warning) to 1.51, after which the message carried alert level 0 (Normal), and the dashboard changed from Critical to Normal. The main loop kept a steady cycle of about 2 s during repeated failed MQTT connection attempts. Wokwi runs slower than real time and does not model real power draw or sensor noise. After the decision logic was moved into a separate header and the 24-phase, persistence-6 settings were adopted, the revised sketch was run again in

Wokwi in the short demonstration mode (three calibration phases, persistence one). A smoothed drift of 1.66 gave no alert; a drift of 1.94 gave a Warning marked as sustained, the published message carried alert level 1, and the dashboard showed Warning, later Critical, and the stale-data banner after 77 s without a message. The calibration messages and the Critical text were not captured in that log, and recovery to Normal, door exclusion, alert tones, the gas alert and alert level 0 were not repeated after the change. The full settings were not stepped through by hand in the simulator; they were run on the board with generated test signals (Section 6.7). A later Wokwi run used a test mode in which the sketch generates the temperature itself, with the short settings. Calibration finished with a baseline of 10.67 s ON and 13.33 s OFF, the first Warning came at a smoothed drift of 1.81 and the Critical alert at 3.71, the door was opened and closed twice with the cycle after the first opening ignored, and the log shows one boot and no reset. An earlier run of the same mode rebooted the chip with a task watchdog error, which is described in Section 6.7.

6.4 Measurements on a Real ESP32-S3

Two tests were run on a real ESP32-S3-N16R8 board (16 MB flash, 8 MB octal PSRAM), both with synthetic inputs. In the first, the cycle-evaluation logic with the rolling window was timed over 20,000 synthetic cycles at 240 MHz: mean 2.03 µs per call (minimum 0.22 µs, maximum 6.81 µs), with 275,808 bytes of program memory used by the benchmark sketch. This covers the decision logic only, not sensors, buzzer, Wi-Fi or MQTT, and no energy was measured. In the second, a phone hotspot and a public MQTT broker were used to measure how network conditions affect the main loop (nominal period 2000 ms). With the MQTT connect call in the main loop, an unreachable broker stalled the loop repeatedly (usually about 3 s, sometimes about 15 s), and the longest loop period was 17,127 ms. With a reachable broker the loop stalled once, for about 1 to 1.5 s. When the network work was moved to its own task, the loop stayed at 2000 ms in both cases. Calling the MQTT server setup only when Wi-Fi was up at boot also meant that MQTT never started if Wi-Fi came up later (no messages in about 530 s), and calling it unconditionally fixed this. The final firmware uses this structure and no longer has the fixed delay or the blocking buzzer routines; its alert latency was not measured separately.

The recorded REFIT phases were also replayed through the decision logic on the board. The data (66,568 phase entries from the four appliances) were stored in flash and run through the same header that the firmware uses, for the original settings and for the proposed ones, and the board printed the number of evaluations at each alert level and a hash of the alert sequence. All eight results (four appliances, two settings) were identical to a Python computation, including the hash. The replay took 16 to 18 microseconds per phase, including the replay loop and flash reads; this is not comparable with the 2.03 microsecond figure above, because the shared logic uses double-precision arithmetic so that it matches the analysis exactly. This shows that the firmware logic reproduces the analysis on real recorded phases. It is not a test with a real fridge or live sensors.

As a separate check on what the chip can run, an Isolation Forest was exported and scored on the ESP32-S3 (Section 4.6). The forest used for this test was trained on synthetic phases, so that the export could be tested on the board before the REFIT model was used. The scores computed on the board agreed with scikit-learn to within 3.2e-8, no alarm decision differed over the 3,000 test vectors, and scoring one vector took about 100 microseconds. This shows that the exported model runs correctly on the chip. It says nothing about detection quality, which is compared in Section 6.6.

6.5 Replay on Household Fridge Traces

Table 3 shows the firmware rule on the healthy traces, with the baseline recalibrated after outages longer than one hour. Alerts were frequent: Warning episodes came every 1 to 6 days (4.8 to 27.5 per 30 days) and Critical episodes ranged from 0.3 to 5.5 per 30 days. The simulation had suggested almost none. With the baseline set once and never recalibrated over 1.5 to 2 years, 31.0%, 0.9%, 15.9% and 15.3% of windows were above 1.8 for the four appliances in the order of the table, so the result also depends heavily on the three cycles used for the baseline. In House 15 that baseline happened to be favourable.

Appliance	Days of data used	Windows above 1.8	Warning / Critical episodes per 30 days
House 2 fridge-freezer	450	8.4%	13.3 / 1.7
House 15 fridge-freezer	478	2.7%	4.8 / 0.5
House 20 fridge	412	14.0%	27.5 / 5.5
House 20 freezer	412	3.4%	5.5 / 0.3

Table 3. The firmware rule (3-phase calibration, 5-phase mean window, Warning 1.8, Critical 3.5) on healthy REFIT traces. Fridge and freezer plugs in three homes; Issues rows kept.

Table 4 shows the leave-one-house-out result. In every held-out house the chosen variation had fewer false alarms than the firmware setting and still reached Critical on an injected 3x fault in every slice, with delays of 5.2 to 6.3 days against 2.9 to 4.9 days for the firmware setting. The firmware setting itself found the 3x fault at Critical in only 67 to 100% of slices. Detection at Warning was unchanged in two cases and about one day slower in the other two. The same direction held when the Issues phases were dropped, and with a minimum phase length of 300 s the results changed very little. The setting best across all three houses, which is a description and not a test, was a calibration of 24 phases, a window of 5, a mean and persistence 6.

Held-out appliance (setting chosen without it)	Warning / Critical episodes per 30 days: firmware, then chosen	Critical found for a 3x fault: firmware, then chosen (median delay)
House 15 fridge-freezer (24 phases, window 5, mean, persistence 6)	4.8 / 0.5, then 1.4 / 0.1	83% (4.9 d), then 100% (5.7 d)
House 2 fridge-freezer (6 phases, window 5, median, persistence 3)	13.3 / 1.7, then 6.1 / 0.6	67% (2.9 d), then 100% (5.5 d)
House 20 fridge (24 phases, window 5, mean, persistence 6)	27.5 / 5.5, then 10.3 / 0.7	100% (4.1 d), then 100% (5.2 d)
House 20 freezer (24 phases, window 5, mean, persistence 6)	5.5 / 0.3, then 3.8 / 0.0	83% (3.5 d), then 100% (6.3 d)

Table 4. Leave-one-house-out comparison of the firmware setting with the setting chosen on the other two homes. The House 20 fridge and freezer were held out together.

The improvement is partial. The House 20 fridge still raised about 10 Warning episodes per 30 days and House 2 about 6, so on these domestic appliances the rule remains noisy. A longer calibration also has a cost, because a fault present during calibration would be absorbed into the baseline. The chosen variation is now the firmware default and was replayed on the board (Section 6.4), but it has not been tested with a real fridge, and the injected faults are artificial. It was selected on three homes of recorded traces.

6.6 Detector Comparison and Warning Threshold

Table 5 gives the averages over the four held-out appliances for the three detectors. All three found every 5x fault. The Isolation Forest had the most false alarms at both targets and the worst result was on the House 20 fridge (16.2 and 28.8 episodes per 30 days), and at the stricter target it missed one of the 3x slices on House 15. The exponentially weighted version had about the same false alarms as the ratio rule at the strict target and fewer at the looser one (3.9 against 6.6), but its threshold carried over badly to House 2 (8.3 and 9.5 episodes per 30

days against targets of 2 and 5), so it was not adopted. On these traces the forest was not better than the ratio rule. The thresholds that the ratio rule picked on the training houses ranged from 1.60 to 2.45, which brackets the 1.8 used in the firmware. The targets were met on the training houses and not always on the held-out one (6.1 episodes on the House 20 fridge against a target of 2), which shows how much the houses differ. These are four appliances in three homes, so the comparison is a rough one.

Target on training houses (episodes per 30 days)	Detector	Healthy episodes per 30 days	3x fault found	Median delay, 3x fault	5x fault found
2	Ratio	2.7	100%	3.1 d	100%
2	EWMA	2.6	100%	3.0 d	100%
2	Isolation Forest	6.4	96%	3.4 d	100%
5	Ratio	6.6	100%	2.1 d	100%
5	EWMA	3.9	100%	2.0 d	100%
5	Isolation Forest	12.5	100%	2.0 d	100%

Table 5. Three detectors on the REFIT traces, leave-one-house-out, averaged over the four held-out appliances. Each detector has one alarm threshold chosen on the training houses to meet the target.

Table 6 shows the effect of moving the Warning threshold. Going from 1.8 to 2.0 removes roughly a third of the false Warning episodes (29 to 39% across the four appliances). The cost falls mostly on mild faults, since 1.5x faults are found in fewer slices, and on a few tenths of a day of delay for a 3x fault. Every threshold from 1.3 to 2.6 still found the 3x fault in every slice. The threshold of 1.8 was kept as a middle value; 2.0 is just as defensible if fewer Warnings matter more than catching mild drift, and only real fridge data can settle that. Moving the threshold does not remove the noise: at 2.2 the House 20 fridge still had about 5 Warning episodes per 30 days.

Appliance	Healthy episodes per 30 days (1.6 / 1.8 / 2.0 / 2.2)	1.5x fault found (1.8 / 2.0 / 2.2)	3x fault median delay (1.8 / 2.0 / 2.2)
House 2 fridge-freezer	7.3 / 4.9 / 3.4 / 2.5	83% / 83% / 83%	1.8 d / 2.2 d / 2.5 d
House 15 fridge-freezer	2.0 / 1.4 / 1.0 / 0.8	100% / 83% / 67%	2.8 d / 3.1 d / 3.6 d
House 20 fridge	14.1 / 10.3 / 6.9 / 5.2	100% / 83% / 83%	1.6 d / 1.9 d / 1.9 d
House 20 freezer	6.3 / 3.8 / 2.3 / 1.6	100% / 100% / 100%	2.4 d / 2.8 d / 3.4 d

Table 6. Warning threshold against false alarms and fault size (24 calibration phases, persistence 6), descriptive over all three homes.

6.7 Whole Firmware on the Board with Generated Test Signals

With a compile-time switch set, the sketch generates its own temperature, door state, gas reading and clock, so the complete firmware can run on a bare ESP32-S3. The generated compressor cycles with a 10 s ON phase and a 12 s OFF phase (with a few percent of jitter), and one scenario is played: healthy cycling through the whole

calibration, a fault that stretches the ON phase to five times its normal length over 12 cycles, recovery to healthy cycling, a gas reading above the threshold for 30 s, and the door open with the temperature held at 9 °C for 3 minutes. The firmware was run with the full settings (24 calibration phases, persistence 6) for about 36 minutes of sketch time, with a program size of 409,869 bytes (31%). Table 7 lists the events from the Serial log, using the sketch's own clock, which starts at 12:00:00.

Clock	Event in the log
12:09:24	Calibration complete after 24 ON and 24 OFF phases (baseline ON 10.92 s, OFF 12.58 s)
12:13:40	Drift 1.82, above the Warning threshold: 1 of 6 evaluations in a row, no alert yet
12:16:14	Warning after six evaluations in a row (drift 2.58)
12:21:12	Critical (drift 4.21); the largest drift in the run was 4.54
12:25:50	Back within normal limits (drift 1.71)
12:27:46	Gas alert raised; cleared at 12:28:16
12:28:34	Door opened
12:28:44	Exposure alert, with the limit set to 2 minutes in this mode (30 minutes in normal use)
12:31:30	Door closed; the next cycle was ignored; no Warning or Critical afterwards

Table 7. Events in the Serial log of the whole firmware on an ESP32-S3 with generated test signals and the full settings.

The log has one boot line and no watchdog message. The Warning came about 2.5 minutes after the drift first crossed 1.8, which is the persistence requirement at work. This run covers calibration with the real settings, persistence, both alert levels, recovery to Normal, the gas alert, the door exclusion and the exposure counter on the real chip. It does not cover real sensors, because none were connected. Wi-Fi did not connect in this run, so MQTT publishing was not exercised on the board, and the run gives no evidence about the LEDs and the buzzer. The exposure limit was shortened, so the 30-minute value was not exercised. Two problems were found on the way. With the network task at priority 1, a long wait on a silent MQTT broker starved the idle task and the task watchdog reset the chip in Wokwi; the task now runs at the idle priority and the MQTT socket timeout is 3 s, and the 36-minute board run showed no reset, which is consistent with the fix but does not prove it, since the board run had no Wi-Fi. The first version of the generated scenario also only grew the fault and never returned to healthy, so it could not show recovery to Normal, and a recovery stage was added.

6.8 Labelled Records of Ultra-Low-Temperature Freezers

The household and generated tests use artificial faults. To see the rule on real compressor faults, it was run on the labelled ultra-low-temperature (ULT) freezer dataset of Huang et al. (Scientific Data, 2023; data released under CC0). Ten Revco freezers of a biobank, held at -80 °C, were used. They were logged once a minute for 6 to 11 years each, and the dataset lists 22 faults with the date of the service report. The controller's State code was used as the compressor state: the compressor counted as ON when either of the two cascade stages was running, and readings with the door-open bit set were skipped. Steady blocks were cut at gaps over 5 minutes or when the chamber was warmer than -65 °C. The firmware rule was applied with its default settings (24 calibration phases, five-phase window, persistence 6, Warning 1.8, Critical 3.5), and alarms less than 3 days apart were

counted as one episode. Healthy time leaves out 90 days before to 30 days after each fault date. The method was written down before the ten freezers were run. The thresholds of 1.5 and 2.0 were added to show sensitivity and were not chosen from the results. Compressor and refrigerant-leak faults (14 with enough data) were treated as the main test, and the other 7 labels (door, wiring, battery, setpoint, condensation, filter) were kept apart. One label had no data.

Threshold	Healthy episodes per 30 days	Compressor or refrigerant faults with an alarm in the last 30 days	Same, last 90 days	Other faults with an alarm, last 90 days
1.5	0.31 (336 episodes)	4 of 14	6 of 14	5 of 7
1.8 (firmware)	0.15 (158 episodes)	2 of 14	4 of 14	2 of 7
2.0	0.11 (114 episodes)	1 of 14	1 of 14	0 of 7
3.5 (Critical)	0.05 (52 episodes)	0 of 14	0 of 14	0 of 7

Table 8. The drift rule on ten ULT freezers (32,355 days of healthy time, 21 labelled faults with data). Faults are dated at the service report.

Three results stand out. First, the rule alarmed rarely: 158 Warning episodes in about 89 freezer-years, which is about 1.8 per freezer per year. This count is an upper bound on the true false-alarm rate, because the unlabelled time is not clean. One freezer (806026) accounts for 31 of the 52 Critical episodes, and its log shows repeated failures to reach the setpoint and several power interruptions. Second, the rule matched the labelled faults poorly. With the firmware threshold, 4 of 14 compressor and refrigerant faults had a Warning in the 90 days before the service date and none reached Critical. Several compressor faults left no trace in the ratio at all: the highest drift in the 90 days before the four stage-2 compressor services on freezers 806024, 806026 and 806033 was 1.02 to 1.21. The rule is also not specific, since 2 of 7 other faults (a door adjustment and a condensation fault) had a Warning as well. Lowering the threshold to 1.5 caught 6 of 14 but doubled the false alarms. The dates are service-report dates and may be well after the fault began or unrelated to the cycling, which limits what these counts can show.

Third, there were 19 long Warning episodes (14 days or more) in the unlabelled record. For the 18 with an events file, the freezer's own log shows a warm alarm, a shutdown, a warm interstage warning or a failure to reach the setpoint within 10 days after the episode ended in 12 cases, and nothing in 6. For example, freezer 806016 had a drift above 1.8 for 81 days (peak 5.5) that ended on 16 August 2018 with a warm alarm and a shutdown, and freezer 806272 had one for 103 days ending the day before its labelled stage 1 compressor service. These events are not fault labels, and an episode also ends when the chamber warms and the block restarts, so they are not counted as detections. They do suggest that, in several freezers, the ratio rose for weeks or months before the freezer's own alarm. The test also differs from the firmware in two ways: the compressor state came from the controller, not from an 8 °C and 2 °C temperature rule, and ULT cascade freezers are not vaccine refrigerators. The conclusion is limited to this: on real freezer records the rule seldom raises alarms and often rises ahead of the freezer's own warm alarm, but it does not reliably match service-dated compressor faults.

Figure 4 shows the drift ratio of each of the ten freezers over the whole record. The orange line is the Warning threshold (1.8) and the red line is the Critical threshold (3.5). Dashed vertical lines mark labelled faults, black for compressor or refrigerant faults.

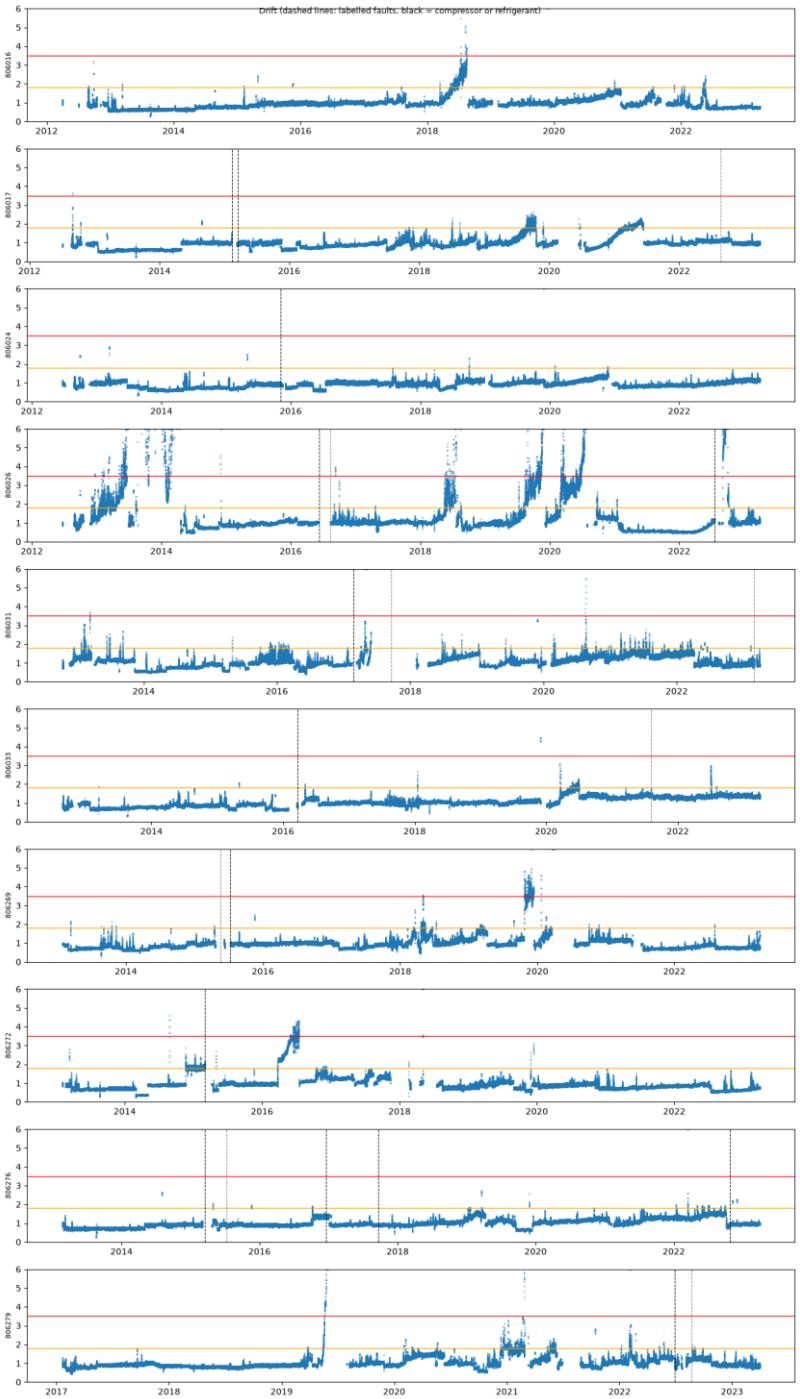

Figure 4. Drift ratio of ten ultra-low-temperature freezers over 6 to 11 years each, with the firmware thresholds and the labelled faults. Data from Huang et al. (2023), CC0.

7. Discussion

7.1 Key Findings

- A predictive fault-detection method tested on genuine industrial failure data (81 of 83 failure hours flagged on held-out data, with the result depending on how a compressor cycle is defined) points to a physically interpretable signal, the compressor on/off cycling ratio. In a vaccine-refrigeration simulation this ratio separates healthy from degrading behaviour clearly (Section 4.4), whereas the Isolation Forest scores on hourly features separate them only weakly (ROC AUC 0.64). On recorded household-fridge traces, however, the rule with its initial settings raised many false alarms (Section 6.5), so the thresholds taken from the simulation did not transfer.
- Single-cycle comparisons of this signal are unreliable when door events occur: in the simulation, one ON/OFF pair exceeded the Warning threshold in 18% of healthy cases, against 2% for a five-cycle average and none once door-affected cycles were left out. A rolling average and a door-event exclusion are therefore both needed.
- The Isolation Forest, which was useful offline for finding the signal, did not turn out to be a better detector than the ratio rule on recorded household-fridge cycling: with the same calibration and persistence it gave more false alarms at both targets (Table 5). The rule is also easier to inspect, and it needs two baseline numbers per unit where the forest needs 50 stored trees.
- On labelled records of ten real ULT freezers the rule rarely raised alarms but matched few of the service-dated compressor faults (Table 8), so the thresholds tuned on a simulation and on household traces cannot be called validated for real faults. The long drifts that ended in the freezers' own warm alarms are the most encouraging observation, and also the least certain, because they come from unlabelled records.
- Deliberately excluding the machine-learning model itself from the embedded device, in favour of a simple, explainable rule set derived from the analysis, is a defensible and common engineering pattern for resource-constrained deployment targets, and does not diminish the validity of the underlying research.
- Designing the safety-critical detection and alerting functions to be fully independent of network connectivity is not merely a convenience but a direct requirement given the documented power and connectivity constraints of the target deployment context.

7.2 Limitations

The following limitations are stated explicitly and are treated as scope boundaries for the current phase of work, not as concealed weaknesses:

- No public dataset of real vaccine-refrigerator failures currently exists; the threshold values used in the embedded system are derived from a domain-specific simulation rather than from real vaccine-fridge sensor logs. Addressing this gap is itself a contribution of the identified research need (Section 2.4).
- The real-world dataset (MetroPT-3) comes from a different system, a metro train's air compressor. It tests the detection method on genuine failure data and is not direct evidence of vaccine-fridge performance. The result depends on how a compressor cycle is defined (81 of 83 failure hours flagged with a 5.5 A definition, 22 and 24 with the other two), the failure windows were not independently verified, and about half of the flags in the held-out test were false.
- The MQ2 gas sensor detects combustible gases and vapours only, and should not be characterised as a general-purpose food-detection sensor; its scope is intentionally limited to a realistic subset of misuse cases.
- Cumulative vaccine-exposure tracking currently resets on a daily cycle rather than tracking exposure across a vaccine's full storage lifetime, which is a simplification appropriate to this prototype stage.

- The public MQTT broker used for the IoT demonstration is unauthenticated and unsuitable for production deployment; a private, authenticated broker would be required in a real installation.
- The complete system was verified in a hardware simulator (Wokwi) and, in a test mode with signals generated by the sketch, on a real ESP32-S3. On the board, the decision-logic timing, the effect of network conditions on loop timing, a replay of recorded household-fridge phases through the decision logic, an Isolation Forest and the whole firmware on generated signals have been run; the inputs were synthetic or recorded, not live sensors. Testing with physical sensors on a physical refrigerator has not been done, and energy use and sensor-to-alert latency have not been measured. The board run of the whole firmware had no Wi-Fi connection, and its exposure alert used a 2-minute limit instead of 30.
- The Critical threshold (3.5 times the baseline) is above anything the simulation produces, so it is a margin for more severe degradation and not a value taken from data. It was exercised in Wokwi with manually driven cycles.
- The replay on household fridge traces (REFIT) uses domestic appliances, has no door information, and tests detection with artificial faults injected into the durations. The calibration and persistence changes were chosen by leave-one-house-out testing on only three homes, they reduced false alarms but did not remove them, and they are now the firmware default but untested with a real fridge.
- On the simulation, the Isolation Forest separates healthy from degrading hours only weakly (ROC AUC 0.64). The device does not use this model.
- In the firmware, the compressor state is inferred from the 8 °C and 2 °C thresholds. A real refrigerator cycles on its own thermostat, so a real build would need a compressor current sensor or a temperature-slope detector.
- In the household-trace and generated-signal tests the faults are artificial, so the detection results describe the detector on injected faults. The ULT freezer records contain real faults, but they are dated at the service report, the compressor state came from the controller and not from temperature, and the freezers are -80 °C laboratory units and not vaccine refrigerators.
- On the ULT freezer records the firmware rule flagged 4 of 14 compressor or refrigerant faults in the 90 days before their service dates at the default threshold, and it also alarmed before 2 of 7 other faults, so it is neither sensitive nor specific for service-dated faults there. The events behind its long unlabelled episodes were not confirmed as faults.
- On the REFIT traces an Isolation Forest on window features did not do better than the ratio rule (Table 5), and moving the Warning threshold reduces false alarms only at the cost of slower or missed detection of mild faults (Table 6).

8. Future Work

- Collection of real sensor data from physical vaccine refrigeration units, including documented failure events, to replace simulation-derived thresholds with directly validated ones.
- Testing the full firmware with physical sensors and a physical refrigerator, followed by field testing, to move beyond the simulated prototype and the board-level timing tests.
- Extension of cumulative exposure tracking to track exposure across a vaccine's full shelf life rather than resetting daily, incorporating established time-temperature stability guidance for specific vaccine types.
- Investigation of a properly exported, quantised machine-learning model (for example, via a compact one-class model or a quantised autoencoder) for direct on-device inference, as a successor to the current threshold-rule approach.
- Migration of the IoT reporting layer to a private, authenticated MQTT broker suitable for production deployment, and evaluation of low-power / intermittent-connectivity reporting strategies appropriate to rural deployment conditions.
- Incorporation of a broader-spectrum gas/air-quality sensor to extend misuse detection beyond combustible vapours.
- Measuring the sensor-to-alert latency and the energy use of the final firmware on the board with real sensors.
- Testing the longer calibration and the persistence requirement with a real fridge and temperature sensing, and testing the rule on the Oak Ridge National Laboratory supermarket refrigeration fault dataset, which pairs fault-free baseline tests with six imposed faults, including an open display-case door.

9. Conclusion

This work has presented the design, testing, and prototype implementation of an edge-AI vaccine refrigeration monitoring system, motivated by a documented cold-chain challenge in rural Ghanaian health clinics. A predictive fault-detection method was first tried on genuine industrial failure data, where it performed well on a held-out test with caveats about how a compressor cycle is defined, and was then adapted to a vaccine-refrigerator simulation and to a resource-constrained embedded target. The result is a prototype that combines predictive fault detection, misuse detection, cumulative exposure tracking and optional IoT reporting, verified in simulation and partly measured on a real microcontroller, and designed to keep working under the power and connectivity constraints of its target setting. Replaying the rule on recorded household-fridge traces showed that the thresholds taken from the simulation false-alarm too often on real compressor cycling, and a longer calibration with a persistence requirement reduced this without removing it. An Isolation Forest on window features did not beat the rule on the same traces, and the complete firmware ran through calibration, both alert levels, recovery and the gas and exposure alerts on a real ESP32-S3 with generated test signals. On labelled records of ten real laboratory freezers the rule alarmed rarely, matched only 4 of 14 service-dated compressor or refrigerant faults, and often rose weeks before a freezer's own warm alarm. The limitations are stated above. The most important next steps are testing with physical sensors on a real refrigerator and replacing the simulation-based thresholds with ones measured on real equipment.

Data and Code Availability

All code, logs and results are in the repository github.com/kwadwoadugyamfi82-web/vaccine-fridge-monitor. The REFIT, MetroPT-3 and ULT freezer data sets are public and are not copied into the repository. ML-Validation/README.md says where to download each one.

Declarations

Competing interests. The author declares no competing interests. Funding. This work received no external funding. The ESP32-S3 board used for the measurements was bought by the author. The system was tested in simulation and on the bench with generated signals. It was not installed in a clinic or tested on a vaccine refrigerator.

References

Direct Relief (2026). The Hidden Systems Behind Global Health.

IQVIA. Tip of the Iceberg: Economic and Environmental Impact of the Vaccine Cold Chain.

World Health Organization (IVB). Monitoring Vaccine Wastage at Country Level.

UNICEF Ghana (2022). In Ghana, a Strong Cold Chain Makes All the Difference. (Dokrochiwa Health Centre; UNICEF/Gavi Cold Chain Equipment Optimisation Platform delivery of 2,343 items including 200+ solar-powered units).

Ghana Health Service. Assessments of national cold chain infrastructure and rural CHPS compound conditions.

Harrabi, M., Hamdi, A., Ouni, B., & Bel Hadj Tahar, J. (2024). Real-time temperature anomaly detection in vaccine refrigeration systems using deep learning on a resource-constrained microcontroller. Frontiers in Artificial Intelligence, 7:1429602. DOI: 10.3389/frai.2024.1429602.

Design of a Lightweight Edge-AI System for Predictive Maintenance on ESP32-S3. MDPI Applied Sciences.

INESC TEC / University of Porto. MetroPT-3 Dataset. UCI Machine Learning Repository, dataset ID 791.

Veloso, B., Ribeiro, R.P., Pereira, P.M., & Gama, J. (2022). The MetroPT dataset for predictive maintenance. Scientific Data, 9, 764. DOI: 10.1038/s41597-022-01877-3.

Davari, N., Veloso, B., Ribeiro, R.P., Pereira, P.M., & Gama, J. (2021). Predictive maintenance based on anomaly detection using deep learning for air production unit in the railway industry. 2021 IEEE 8th International Conference on Data Science and Advanced Analytics (DSAA), 1–10. DOI: 10.1109/DSAA53316.2021.9564181.

Sun, J. et al. Test Data of Supermarket Refrigeration System Fault Detection and Diagnostics. Figshare. DOI: 10.6084/m9.figshare.13232627.

Huang, T., Nøstvik, S., Bacher, P., Jensen, J.K., Markussen, W.B., & Møller, J.K. (2023). Labelled dataset for Ultra-Low Temperature Freezer to aid dynamic modelling & fault detection and diagnostics. Scientific Data. DOI: 10.1038/s41597-023-02808-6.

Murray, D., Stankovic, L., & Stankovic, V. (2017). An electrical load measurements dataset of United Kingdom households from a two-year longitudinal study. Scientific Data, 4, 160122. DOI: 10.1038/sdata.2016.122.

Murray, D., Liao, J., Stankovic, L., Stankovic, V., Hauxwell-Baldwin, R., Wilson, C., Coleman, M., Kane, T., & Firth, S. (2015). A data management platform for personalised real-time energy feedback. Proceedings of the 8th International Conference on Energy Efficiency in Domestic Appliances and Lighting (EEDAL).

REFIT: Electrical Load Measurements (Cleaned). University of Strathclyde. DOI: 10.15129/9ab14b0e-19ac-4279-938f-27f643078cec.

MetroPT-3 Dataset (Kaggle mirror).

ResearchGate. "IoT-Based Multi-Parameter Monitoring for Robust Vaccine Storage in Underserved Areas."

Jurnal Otomasi Kontrol dan Instrumentasi. "Design of a Temperature and Humidity Monitoring System for Cold Chain Vaccine Storage."

GitHub (MananDesai28). "Vaccine-Temperature-Monitoring-System-using-IoT."

"Smart IoT-Based COVID-19 Vaccine Supply Chain, Monitoring, and Control System."

DFRobot. "TinyML-ESP32 Integration" for predictive maintenance.

"Embedded TinyML for Predictive Maintenance: Vibration Analysis on ESP32."