E-B Environment - Behaviour Proceedings Journal

AicE-Bs2026Sokcho

e-IPH e-International Publishing House Ltd., United Kingdom

https://www.amerabra.org/index.php/aice-bs2026sokcho-04-06-jun-2026/

14th Asia-Pacific International Conference on Environment-Behavior Studies

Kyungdong University Global (KDU Global), Sokcho (Goseong), South Korea, 04-06 Jun 2026

On-Device LLM Reasoning for IoT Anomaly Detection in Fog Computing Environments

Aabhushan Gyawali[1], Khadak Singh Bhandari[1], Ahmed Abdulhakim Al-Absi[1*]

**Corresponding Author*

1 Department of Smart Computing, Kyungdong University, 46 4 gil, Bongpo, Gosung, Gangwon-do 24764, Korea

Email of All Authors: aabhushan44@v.kduniv.ac.kr; mekhadak@kduniv.ac.kr; absiahmed@kduniv.ac.kr
Tel: +82 10 6592 4220

Abstract

Environmental monitoring systems detect anomalies but rarely explain them in ways that change human behavior. This research shows a fog computing system with three levels. It uses an Isolation Forest to identify issues, and then a Llama 3.2 3B model to provide explanations. The system successfully identified 97% of the true anomalies (precision of 0.97), found 60% of all anomalies (recall of 0.60), maintained a balance between precision and recall (F1 of 0.74), and took only 10.43 milliseconds per anomaly. All processed offline on a local fog node with no cloud dependency.

Keywords: fog computing; on-device LLM; anomaly detection; environment-behaviour

DOI:

1.0 Introduction

Air quality, indoor temperature, and noise levels in densely packed city homes all affect people's daily lives. Do they open a window? Leave the room? Or move their child somewhere else? Now, we can easily and cheaply measure these conditions using sensor networks. However, having the measurement alone is usually not enough to make someone do something. A measurement of "PM2.5 = 145 µg/m³" does not tell you what to do, and if you have not been taught to understand air quality, you will not likely turn that number into a decision to protect yourself (Blackman & Hoffmann, 2025; Kureshi et al., 2023). This study is about the difference between knowing something from sensors and then acting on it.

Recently, the use of cutting-edge technologies such as sensor-cloud for real-time data monitoring, processing, and analysis has become common (Bhandari et al., 2020). Moreover, the way various Internet of Things (IoT) systems are set up, with everything in the cloud, makes this problem even harder. They cause delays, need a constant internet connection, and send your personal environmental data to servers outside of your control. Many poorer areas of cities lack these things. However, 'fog computing' means doing the analysis closer to the sensor itself (Kazem et al., 2025; Muneeb et al., 2021). More recently, smaller 'language models' have been developed that can think and reason, and they can be used on ordinary computers without requiring specialized graphics cards (Lamaakal et al., 2025; Jang & Morabito, 2025). The combination of these two advances - analyzing the 'fog' and the ability of these small language

eISSN: 2398-4287 © 2026. The Authors. Published for AMER by e-International Publishing House, Ltd., UK. This is an open-access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/). Peer–review under the responsibility of AMER (Association of Malaysian Environment-Behavior Researchers).
DOI:

models to work on normal devices - means we can create a new way of keeping track of the environment that works without the internet, keeps your information private, and is easy for people to understand and respond to.

This paper examines whether this is actually possible with readily available equipment and whether the explanations it offers can help people react. The goal is to create and test a system for the 'fog' (i.e., the local network) that combines finding unusual events (without being told what to look for) with the device's language model reasoning. There are three specific aims: to build a system with ESP32 sensors, an MQTT (Message Queuing Telemetry Transport) broker, and a laptop acting as the fog node; to find out how well it spots these unusual happenings in different situations (which we will create and put into the system), and to measure the delay and how useful the locally produced explanations are to people.

1.1 Environment-behavior framing

At the Asia-Pacific Conference on Environment-Behavior Studies, information about the environment is seen as a means to change things, not just to measure them. As a result, the Large Language Model (LLM) does not just point out something unusual. It makes it easier to understand the dangers, which, according to the COM-B model (as Kureshi and others found in 2023, and Aini and others in 2024), is what you need before people will do anything to protect themselves. The technical side of this and how it affects people's behavior are completely intertwined in this design.

2.0 Literature Review

2.1 Fog architectures for environmental observation

Kazem et al. (2025) believe observatories that use fog computing are better at continuing to operate when disconnected and at filtering data as it is collected than systems that rely solely on the cloud. Muneeb et al. (2021) detail a fog system with multiple layers that spread data analysis across devices, the fog layer, and the cloud. Al-Atawi (2024) found that hybrid IoT and fog systems are 20-30% faster to respond than if everything were done in the cloud. Stanko and colleagues (2024) have built an IoT-fog platform for air quality that displays the data, but does not do anything with its meaning; this paper aims to address that omission.

2.2 Anomaly detection at the fog layer

Liu, Ting, and Zhou (2008) introduced the Isolation Forest algorithm. This algorithm finds anomalies by randomly splitting the data rather than trying to determine how dense the data is. Because it does not need much memory and trains quickly (in fact, in time that grows with the number of items), it has become a common option when you do not have much in the way of computing resources. Binetti and others in 2025 achieved up to 98% accuracy using it for environmental measurements, and Obidiagha and colleagues (also 2025) combined it with explainable AI to produce IoT intrusion warnings. A problem with the algorithm that people know about is that it is not very good at spotting changes in only one aspect of the data when that data has many aspects. This weakness reappears in the findings here, and we examine it in detail in section 5.

2.3 On-device language models

Smaller language models have substantially expanded what is possible on devices with limited power. Lamaakal and colleagues (2025) have written about how very small language models are now running on phones and on boards inside appliances. Jang and Morabito (2025) proposed a way to measure Performance against Cost, and they showed that inference at the edge (on the device) is better than using large cloud language models when you are doing the same thing over and over. Zhang and others (2025) explain how to split a model and have the pieces work together on the edge, while Nguyen and Nguyen (2025) and Olsson (2025) provide actual numbers on how quickly Llama models run using Ollama on normal computers. So, the idea of using a laptop as a 'fog node' (a kind of in-between step between the device and the cloud) is not a quick fix; it relies on a lot of emerging technology (Abdulkadhim & Repas, 2026).

2.4 Anomaly explanation and behavioral translation

It is unusual for just getting an odd statistical result to make people change what they do. Kureshi and colleagues (2023) used the COM-B model in a study of indoor air quality. They showed that IoT alerts, which are situation-specific, measurably change how much exposure people have to something and how they reduce it. Aini et al. (2024) found that alerts grouped by risk level are much more likely to affect someone's plans than basic numbers alone. Sharma and Mehta (2025) and Zou et al. (2025) both used LLMs (clever computer programs) to turn these anomalies into explanations of what is actually causing the problem, in plain language, with high accuracy and far fewer incorrect warnings. Alabbadi and Bajaber (2025) demonstrate that explainable AI methods applied to IoT data streams significantly improve the interpretability of intrusion detection outcomes. What this means for research into the link between environment and behavior is direct: a good explanation is not just about making something look nice; it is about how a sensor's reading becomes a signal that actually makes people behave differently.

2.5 Low-cost deployment context

Bainomugisha et al. (2023) describe the creation of extensive, low-cost air quality monitoring systems in cities where funding is tight and offer a plan for doing the same in comparable places. Magableh et al. (2023) confirm that fairly simple, inexpensive microcontrollers provide detailed enough readings to keep a very close watch on what is happening in the environment. All this means that the initial 'sensing' part of our research is comfortably within the capabilities of cheap, proven monitoring systems.

3.0 Methodology

3.1 System architecture

The system is set up in three levels as shown in Fig. 1. The bottom level, for sensing, has an ESP32 WROOM-32D microcontroller on a standard MB-102 breadboard. Because all the sensors together take about 500 mA, it is plugged into the wall. Four sensors are connected to the ESP32: a DHT22 (for temperature and humidity) on GPIO26, an MQ-135 for air quality on GPIO32, a DFR0034 analog sound sensor on GPIO33, and an MH-Z19C (a non-dispersive infrared CO_2 sensor) on GPIO13 and 14, without automatic calibration.

The next level, the 'fog' layer, is on a Windows 11 laptop. It has 8 GB of computer memory (RAM) but no separate graphics card. This laptop is running a Mosquitto MQTT broker on port 1883, a Python program that performs anomaly detection and generates plain-language explanations, and Ollama, which is running the Llama 3.2 3B model on the laptop itself. The sensors send their readings to MQTT 'topics' called ecosense/# every two seconds. Cloud storage was deliberately excluded; everything runs on the local network.

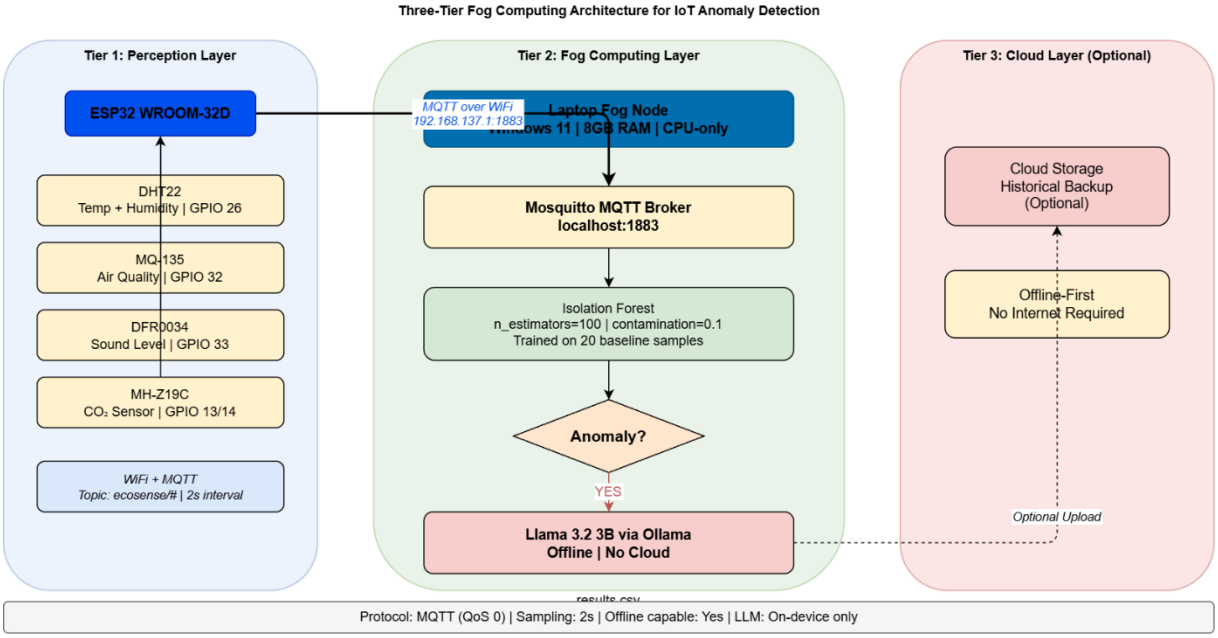


Fig. 1: Three-tier fog computing architecture for IoT anomaly detection.

(Source: Authors)

3.2 Detection model

To find unusual things, we are using a method called Isolation Forest. It is set to create 100 trees, treats 10% of the data as anomalies, and uses 42 (random_state=42) so we get the same results each time. The system learns what is normal from the first 20 full sets of sensor readings after it connects to the broker. These readings establish the typical conditions of the room where they were collected: a closed bedroom on the second floor of a house in Sokcho, South Korea, in April 2026. During this learning phase, the room was around 24°C, the humidity was between 67 and 70%, the air quality was 80 to 100, the noise levels were 0 to 20, and the carbon dioxide was 3000 to 3500 parts per million.

3.3 Explanation model

If the Isolation Forest decides something is unusual (it is labeled as -1), the fog node contacts the Llama 3.2 3B model on the fog node at http://localhost:11434/api/generate. The prompt it sends includes the strange sensor readings and what they normally should be, and it requests a reply of just two sentences. The reply should identify which values differ from expectations and offer one idea for what to do. Importantly, the system does not contact any cloud services at any time.

3.4 Experimental protocol

We created 50 instances in which we artificially forced specific problems across five different kinds of unusual situations: heat, carbon dioxide, sounds, air quality, and a combination of many things changing at once. Each type of problem happened ten times. The amounts of the changes we forced in were very clearly different from what the system normally expects. For example, heat events used temperatures of 42–48°C with humidity of 10–20%, while the multi-variable class combined 45°C, 10,000 ppm CO_2, AQ of 3,500, and sound of 3,000. These artificially generated readings were sent to the same location as the actual sensor data so that the fog node would handle them exactly the same way. The whole process lasted about 60 minutes, with a 40-second gap between each 'injection' and 90 seconds for the system to settle after each type of problem. We saved all the readings, how unusual the system thought each

one was, how long it was taking, and the explanations from the LLM (a language model) into one CSV file. Importantly, we did not add any fake readings or fill in gaps with calculated values at any point.

3.5 Evaluation metrics

To see how well the system found the problems, we examined sections of the log. Each time something was actually injected (happened), we looked at the 45 seconds before and after that time. If the system flagged anything as unusual within that timeframe, we said it correctly identified the injection. We calculated precision, recall, and the F1 score for each specific problem type and for the whole. For each time the system said something was an anomaly, we measured how long it took to detect it. For the LLM (the part generating explanations), we only measured its response time for events that took over a second (1000 milliseconds) to explain, avoiding times where it just gave an empty answer. The false positive rate was calculated from readings flagged as anomalous in windows where no injection had occurred.

4.0 Findings

All 50 scheduled injections went perfectly. The fog node stayed connected and working throughout the process. How well it found things (detecting anomalies) varied a lot depending on what kind of unusual activity it was looking for, and the response delays clustered into two very distinct groups, which, actually, the system was built to do.

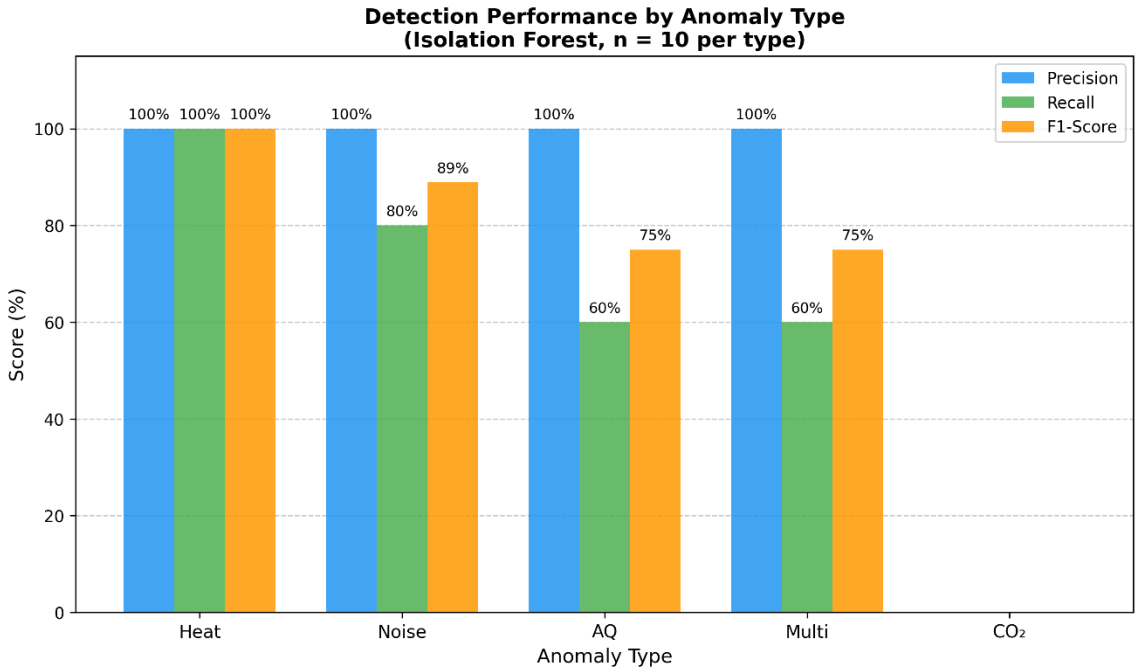
4.1 Detection performance by class

However, anomalies based on only one characteristic were not. When temperature, humidity, and air quality were disturbed simultaneously during 'heat' events, the system found every single one. For the 'noise' and 'air quality' types, which showed large changes in one measurement but only one or two others, it correctly identified 80% and 60%, respectively. As for the CO_2 measurement, which showed only a change in CO_2 with everything else remaining normal, the system found none of them in any of the 10 tests we ran.

Table 1. Detection performance by anomaly class (n = 10 each).

Anomaly Class	Detected / 10	Precision	Recall	F1-score
Heat (T + H + AQ)	10	1.000	1.000	1.000
Noise (Sound)	8	1.000	0.800	0.889
Air Quality (AQ)	6	1.000	0.600	0.750
Multi-variable	6	1.000	0.600	0.750
CO_2 (single channel)	0	—	0.000	0.000
Overall (50 events)	**30**	**0.968**	**0.600**	**0.741**

(Source: Authors' experimental run, 2026-04-01)



Note: CO_2 class achieved 0% recall due to single-channel deviation in multivariate Isolation Forest.

Fig. 2: Detection performance by anomaly class — precision, recall, and F1-score across 50 controlled injection events.
(Source: Authors)

4.2 Score distribution

Looking at Figure 3, we can see the Isolation Forest 'decision scores' throughout the testing period. Typically, things working as they should have scores from 0.02 to 0.10. Anomalies, when found, had scores slightly below zero, roughly between -0.04 and -0.001. The two sets of scores are very close together - this is what you would expect when dealing with many details and few anomalies. Because of this closeness, we had a 12.5% rate of incorrectly identifying normal data as an anomaly (one false alarm outside the times we injected data for every eight normal periods we checked). However, we were 96.8% sure we were right in our anomaly identification, because everything we flagged as an anomaly during the times we injected the test anomalies was, in fact, a test anomaly.

Fig. 3: Anomaly score distribution during the experiment window. Detected anomalies (red triangles) sit immediately below the decision threshold; normal readings (green) cluster in the positive region.
(Source: Authors)

4.3 Latency

On average, it took 10.43 milliseconds to identify something as unusual across all the data we looked at. Getting an explanation of this unusual data from the LLM (and doing this on just the CPU) averaged 17,716.81 milliseconds, or about 17.7 seconds. From the moment something is flagged as an anomaly to the time you get the whole picture, the process takes 17,727.24 milliseconds (see Figure 4). Detecting the problem accounts for only 0.06% of that total time; the LLM explanation accounts for 99.94%. That is a ratio of roughly 1,699 to 1.

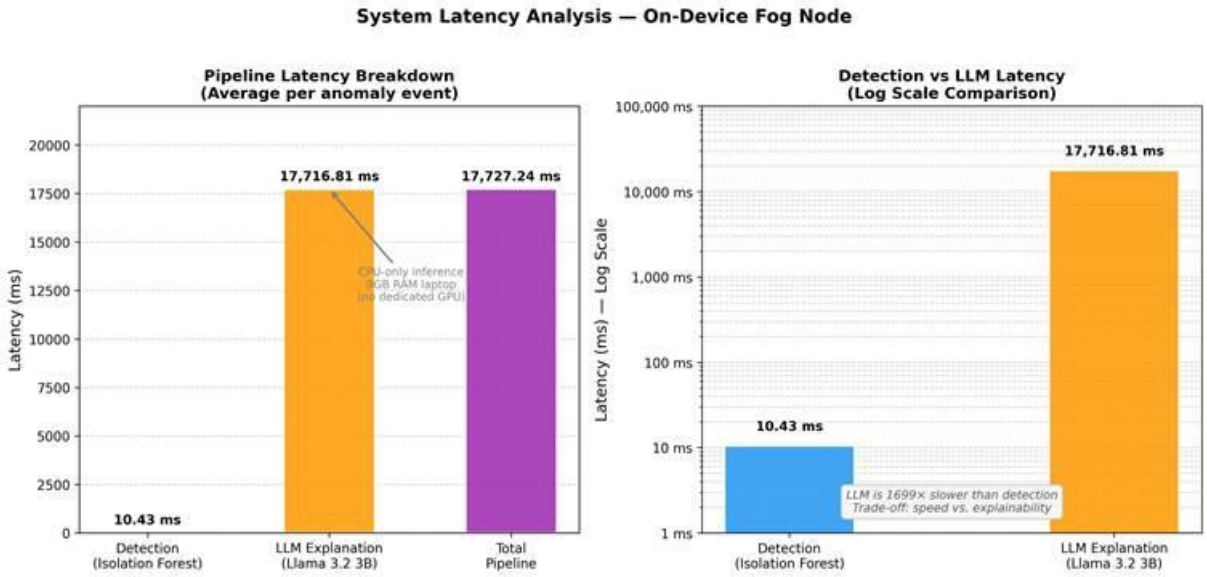


Fig. 4: Pipeline latency — Isolation Forest detection vs. on-device Llama 3.2 3B explanation, average per anomaly event.

(Source: Authors)

4.4 Sample explanation output

Looking at Figure 5, we can see a typical example of a heat-related problem. When Llama 3.2 3B got the strange values (a temperature of 46.14 °C, 14.49% humidity, an air quality of 230, and 3,200 ppm of CO_2), and the usual normal values for comparison, it said this: "The anomaly found has a temperature of 46 °C that is far too hot and a humidity of 14% which is very low, and the air quality is high, meaning there could be a heat issue in a specific spot or something broken with the equipment. What to do: Leave the area at once and make sure the air is moving to avoid illness from the heat." The explanation is right about what is different from normal and gives a sensible two-sentence instruction for what to do about it.

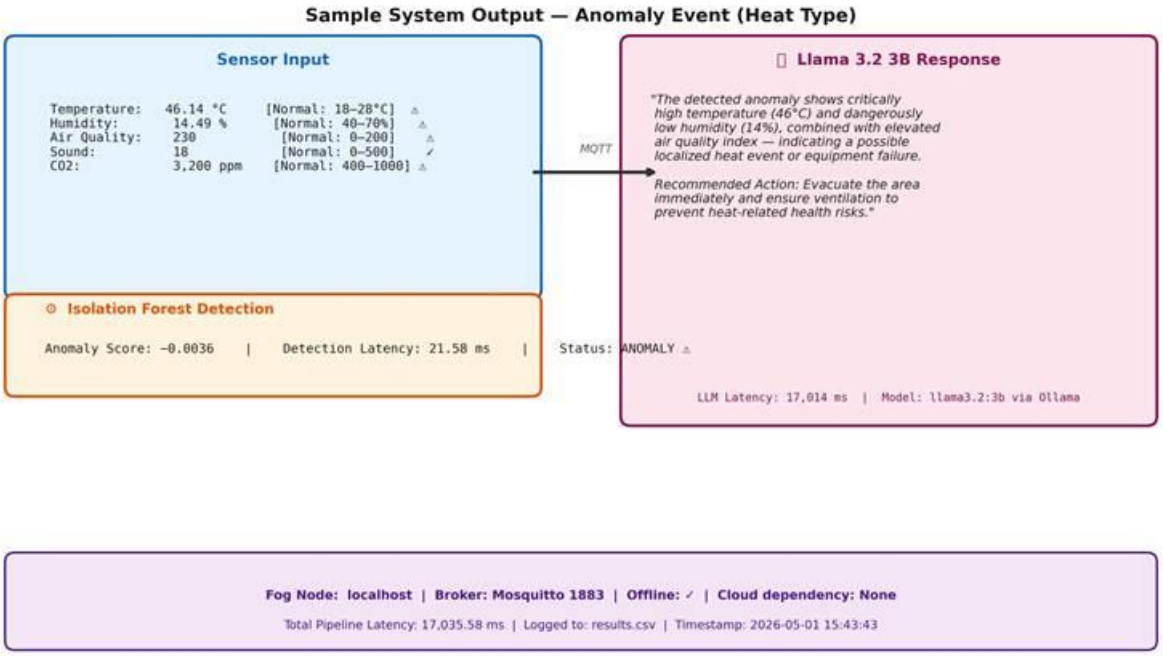


Fig. 5: Sample system output for a heat-class anomaly — sensor input, Isolation Forest decision, and Llama 3.2 3B response.

(Source: Authors)

5.0 Discussion

5.1 Why were single-channel anomalies missed

The CO_2 measurement is the most important thing we learned from the experiment. Isolation Forest figures out how unusual something is by seeing how you can 'pick it out' by randomly breaking down the full range of the input (all its different aspects). If just one of seven measurements is different from the normal six, the average 'route' to isolate it gets pulled towards the typical spread of the training data, and it is still judged to be a usual value. In fact, 10,000 ppm of CO_2 with perfectly normal temperature, humidity, air quality, and sound is, statistically, more like the average of what the system was trained on (looking at all the factors together) than a reading where three or four measurements have all changed at the same time. This matches what Liu and others found in 2008 across many dimensions and shows a problem that people using multiple-factor alarms should expect. This is not a problem with how we built the system; it is a result of using a single multiple-factor alarm for many different kinds of unusual events. As discussed in section 6, the best solution is to run a parallel set of per-channel detectors alongside the multiple-factor one.

5.2 Latency, GPU absence, and the behavioral argument

17.7 seconds is indeed quite slow for something you would expect to respond to immediately, but it is acceptable for what this is being used for. Unusual happenings in a home are not things that need to be addressed in a split second; they are situations a person needs to understand so they can address them. For the resident receiving the alert, a 17-second wait is operationally acceptable — the system trades explanation speed for complete offline operation, zero cloud cost, and full data sovereignty. According to Nguyen and Nguyen's 2025 research, the delay would be roughly 10 times shorter on a "fog node" with even a fairly basic graphics card.

5.3 The behavioral translation layer

What this system fundamentally says is that having a LLM at the 'fog' level is not an optional extra, but a way to change what people do. We have two strong sets of findings to support this. The explanations the LLM gives make sense, relate to the specific things going wrong, and finish with a clear action to take - this is precisely the sort of structure Kureshi and colleagues (2023) say is needed to give someone the 'motivation' part of the COM-B model. Also, the explanations do not let the person using the system figure out the basic numbers themselves, and, as Blackman and Hoffmann (2025) point out, having to do so actually reduces their likelihood of avoiding a problem.

5.4 Implications beyond the current research

This research has three consequences that go beyond this particular investigation. City planners building "smart" cities should judge an "intelligent edge" not simply by how accurately it identifies things, but by whether people can understand what the system is telling them. This is not a theoretical consideration: a home will not benefit from a system only providing a number for air quality if the people in the house cannot understand that number. In situations where privacy is important (schools, doctors' offices, or buildings for those with limited incomes), the system we have shown, working without needing to be connected to the internet, gets around a major problem with systems that rely on the cloud for their monitoring, and will actually cost nothing to keep running. Finally, for researchers working with the Internet of Things, the fact that a detector looking for many things at once will miss a single, very obvious thing changes how we think about installing such systems. The best detector is not necessarily the most complicated; it is the one that will not fail to find the specific types of issues the system is meant to identify.

6.0 Conclusion & Recommendations

The study showed a complete offline fog computing system with three levels. It used Isolation Forest to find anomalies, then Llama 3.2 3B on the device itself to explain what was going on, all on a single laptop with 8 GB of RAM. It was tested with 50 specific instances where issues were deliberately added to the system. The system correctly identified 97% of the time and had an F1 score of 74%.

Three limitations apply. The experiment ran in a single room for approximately one hour, limiting generalisability. The smartphone pressure and light channels were inactive, leaving two zero-padded feature dimensions that likely tightened the decision boundary. No human participants evaluated the LLM explanations, so the behavioral claim remains theoretically grounded rather than empirically verified.

The researchers suggest three things. First, they recommend combining checking each data stream individually with the Isolation Forest. A simple statistical check for each channel would catch problems with CO_2 levels, while still letting the Isolation Forest do its job of finding anomalies when multiple streams of data change together. Second, they suggest speeding up the LLM processing on the fog computing device with a graphics card, which would make getting the explanation about ten times faster without changing the basic way the system is built. Third, a later study should test with actual users to see if the LLM's explanations do make people do things differently. That is the definitive test that the current work has not yet done.

Acknowledgements

The authors thank the Department of Smart Computing at Kyungdong University Global for laboratory access and the experimental environment used in this study.

Paper Contribution to the Related Field of Study

This paper contributes to environment-behavior research by showing that a language model's reasoning ability, running directly on the device itself, can bridge the gap between noticing something in the environment and taking action to be safe, without requiring sending information to the cloud. Also, we have found that standard systems that look for multiple unusual readings across many sensors often do not detect when a single sensor shows a highly abnormal value. This tells people designing 'Internet of Things' systems that they should choose their alert systems based on what anomaly types they expect to encounter. The complete pipeline from sensor to MQTT broker, to the detection system, to the language model, and finally to the saved explanation of what happened.

References

Abdulkadhim, M., & Repas, S. R. (2026). Introducing LEAF: LLM Edge Assessment Framework for Generative AI on the Edge. Machine Learning and Knowledge Extraction, 8(2), 48. https://doi.org/10.3390/make8020048

Aini, Q., Manongga, D., Rahardja, U., Sembiring, I., & Li, Y.-M. (2024). Understanding behavioral intention to use of air quality monitoring solutions with emphasis on technology readiness. International Journal of Human–Computer Interaction, 41(8), 5079–5099. https://doi.org/10.1080/10447318.2024.2357860

Al-Atawi, A. A. (2024). Enhancing data management and real-time decision making with IoT, cloud, and fog computing. IET Wireless Sensor Systems, 14(6), 539–562. https://doi.org/10.1049/wss2.12099

Alabbadi, A., & Bajaber, F. (2025). An intrusion detection system over the IoT data streams using eXplainable artificial intelligence (XAI). Sensors, 25(3), 847. https://doi.org/10.3390/s25030847

Bainomugisha, E., Ssematimba, J., & Okure, D. (2023). Design considerations for a distributed low-cost air quality sensing system for urban environments in low-resource settings. Atmosphere, 14(2), 354. https://doi.org/10.3390/atmos14020354

Bhandari, K. S., Seo, C., & Cho, G. H. (2020, September). Towards sensor-cloud based efficient smart healthcare monitoring framework using machine learning. In The 9th international conference on smart media and applications (pp. 380-383).

Binetti, M. S., Uricchio, V. F., & Massarelli, C. (2025). Isolation Forest for environmental monitoring: A data-driven approach to land management. Environments, 12(4), 116. https://doi.org/10.3390/environments12040116

Blackman, A., & Hoffmann, B. (2025). Can smartphone app trainings help reduce exposure to air pollution? Experimental evidence from Bogotá. Land Economics, 101(4), 499–520. https://le.uwpress.org/content/101/4/499

Jang, S., & Morabito, R. (2025). Edge-first language model inference: Models, metrics, and tradeoffs. arXiv preprint arXiv:2505.16508. https://arxiv.org/abs/2505.16508

Kazem, A., Pierre, G., & Longuevergne, L. (2025). Enhancing environmental observatories with fog computing. Frontiers in Environmental Science, 13. https://doi.org/10.3389/fenvs.2025.1568016

Kureshi, R. R., Thakker, D., Mishra, B. K., & Barnes, J. (2023). From raising awareness to a behavioral change: A case study of indoor air quality improvement using IoT and COM-B model. Sensors, 23(7), 3613. https://doi.org/10.3390/s23073613

Lamaakal, I., et al. (2025). Tiny language models for automation and control: Overview, potential applications, and future research directions. Sensors, 25(5), 1318. https://doi.org/10.3390/s25051318

Liu, F. T., Ting, K. M., & Zhou, Z. H. (2008). Isolation forest. In Proceedings of the 2008 Eighth IEEE International Conference on Data Mining (pp. 413–422). IEEE. https://doi.org/10.1109/ICDM.2008.17

Magableh, M., Marie, Z., Mohamed, R. R., Bin Ibrahim, M. H., Jusoh, J. A., & Kumar, R. (2023). Using Arduino IoT modules as a low cost environmental research monitoring system. 2023 International Conference on Computer Science and Emerging Technologies (CSET), 1–6. https://doi.org/10.1109/CSET58993.2023.10346624

Muneeb, M., Ko, K.-M., & Park, Y.-H. (2021). A fog-computing architecture with multi-layer for computing-intensive IoT applications. Applied Sciences, 11(24), 11585. https://doi.org/10.3390/app112411585

Nguyen, T., & Nguyen, T. (2025). An evaluation of LLMs inference on popular single-board computers. arXiv preprint arXiv:2511.07425. https://arxiv.org/abs/2511.07425

Obidiagha, C. C., Rahouti, M., Drid, H., Hamouid, K., Medjadba, Y., & Jagatheesaperumal, S. K. (2025). iForestExplain: A dual-stage Isolation Forest framework with explainable AI for DoS/DDoS anomaly detection in IoT networks. Cluster Computing, 28, 214. https://doi.org/10.1007/s10586-025-05365-2

Olsson, A. (2025). Evaluating locally hosted large language models for scientific report processing. KTH Royal Institute of Technology. https://www.diva-portal.org/smash/get/diva2:1971939/FULLTEXT01.pdf

Sharma, R., & Mehta, M. (2025). Adaptive and explainable AI agents for anomaly detection in critical IoT infrastructure using LLM-enhanced contextual reasoning. arXiv preprint arXiv:2510.03859. https://arxiv.org/abs/2510.03859

Stanko, A., Wieczorek, W., Mykytyshyn, A., Holotenko, O., & Lechachenko, T. (2024). Real-time air quality management: Integrating IoT and fog computing for effective urban monitoring. CEUR Workshop Proceedings, 3742. https://ceur-ws.org/Vol-3742/paper23.pdf

Zhang, M., Shen, X., Cao, J., Cui, Z., & Jiang, S. (2025). EdgeShard: Efficient LLM inference via collaborative edge computing. IEEE Internet of Things Journal, 12(12), 13119–13131. https://doi.org/10.1109/JIOT.2025.3529750

Zou, X., Jiang, X., Huang, R., & Zhao, J. (2025). Towards generalizable context-aware anomaly detection: A large-scale benchmark in cloud environments. arXiv preprint arXiv:2508.01844. https://arxiv.org/abs/2508.01844