Current Journal of Engineering and Science Research
Volume 3, Issue 11, pp 26-34, November 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.311004
Available Online at https://www.cjesr.com

An Audit of Temporal and Acquisition Confounds in the TON_IoT Telemetry Datasets

Reginald Ekene Ogu

Department of Computer Engineering, Federal University of Technology, Owerri, Imo State, Nigeria
E-mail: reginald.ogu@futo.edu.ng | ORCID: https://orcid.org/0000-0001-9828-8839

Abstract: The TON_IoT telemetry files, covering seven IoT and IIoT devices, are widely used to train and compare attack detectors, and many studies report near-perfect accuracy on these files. This audit asks what those scores measure. Each device file was examined for class timing, sampling behaviour, value encoding and information content. Classifiers were then trained on sensor values, timestamp features and acquisition features under random and within-class chronological splits. Every attack class was recorded in its own time window, and all normal data predate the attacks by three weeks. Timestamp features alone reached a macro-F1 of 1.00 on six devices. Normal readings were burst-sampled, so reading intervals alone separated normal from attack traffic with binary F1 of 0.99 or higher on three devices. In the garage-door file, one field is encoded as true/false for every normal row and 0/1 for every attack row. With sensor values only, macro-F1 under random splitting ranged from 0.07 to 0.98, but under chronological blocking the best device fell from 0.98 to 0.46. Three devices carry so little information that no classifier can exceed 38% accuracy. Recommendations for evaluation and dataset maintenance are given.

Keywords: IoT telemetry, shortcut learning, data leakage, temporal confounding, intrusion detection, dataset audit.

I. INTRODUCTION

Most IoT intrusion detection research studies network traffic. A smaller but growing body of work looks instead at the data that devices themselves produce: temperature readings, door states, GPS coordinates and industrial register values [1]–[3]. The idea is appealing. If an attack changes what a sensor reports, or how often it reports, a detector placed near the device might catch it even when the network view is encrypted or incomplete.

The TON_IoT telemetry datasets were released to support this kind of work [1]. The files contain readings from seven devices, namely a fridge, a GPS tracker, a garage door, a Modbus controller, a motion-activated light, a thermostat and a weather station, collected on the UNSW Canberra testbed under normal operation and under eight attack types [4]. Each device has a curated train–test file with a binary label and an attack-type field. Ensemble classifiers [5], [6], transfer learning [7] and a range of other models have been evaluated on these files, and high scores are routinely reported.

High scores invite a simple question that is rarely asked: what did the model use? Recent work on machine-learning evaluation has shown, across many fields, that models exploit whatever regularity separates the classes in a benchmark, whether or not that regularity has anything to do with the phenomenon of interest [8]–[10]. In security datasets the most common culprits are artefacts of collection: who generated the traffic, when it was recorded, and how it was processed [11]–[13]. Telemetry is especially exposed. A sensor reading on its own carries little information, so whatever else comes with it, such as a timestamp, a sampling rate or an encoding quirk, can easily carry more.

This paper audits the seven TON_IoT telemetry train–test files with that concern in mind. It does not propose a new detector. It measures how much of the achievable performance comes from four sources: when a reading was taken, how often readings arrived, how values were written, and what the sensor actually reported. The contributions are:

1. A timeline reconstruction showing that every attack class occupies its own time window on every device, and that the normal data were captured three weeks before any attack.
2. Controlled experiments separating sensor content, timestamp features and acquisition features under random and within-class chronological splits.
3. Identification of two acquisition artefacts: burst sampling of normal data, and a value-encoding difference in the garage-

Copyright © 2026 CJESR All Rights Reserved

Current Journal of Engineering and Science Research
Volume 3, Issue 11, pp 26-34, November 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.311004
Available Online at https://www.cjesr.com

door file that labels every row on its own.
4. An information-ceiling analysis showing that three devices cannot support multiclass detection from sensor values at all.
5. Practical recommendations for researchers using these files and for maintainers of similar datasets.

II. BACKGROUND AND RELATED WORK

2.1 The TON_IoT telemetry data

Alsaedi et al. introduced the telemetry component of TON_IoT and described a testbed in which IoT and IIoT services published readings through MQTT brokers to cloud and fog layers, while Kali Linux machines launched attacks against the same infrastructure [1]. The wider TON_IoT collection also includes network flows and operating-system logs [4], and Booij et al. examined how its heterogeneous features and labels compare with other IoT datasets [14]. Labelling relied on the known addresses and activity windows of the attacking machines [4].

Studies using the telemetry files include stacking and boosting ensembles evaluated per sensor [5], [6] and a transfer-learning approach that treats the seven devices as related domains [7]. Broader reviews of IoT attack detection list TON_IoT among the main benchmarks [15]–[17], alongside Edge-IIoTset [18], X-IIoTID [19] and CICIoT2023 [20].

2.2 Leakage, shortcuts and dataset artefacts

Kaufman *et al.* formalised leakage as information available during training that would not be available in deployment [21]. Kapoor and Narayanan found leakage behind irreproducible results in 17 fields and proposed a taxonomy of its forms [9]. Rosenblatt et al. quantified how much leakage inflates predictive performance in neuroimaging [22], and Whalen *et al.* described similar pitfalls in genomics, including the effect of non-independent samples on cross-validation [23].

Shortcut learning is the behavioural side of the same problem: a model latches onto an easy cue that happens to correlate with the label in the training distribution [8]. In network security, D'hooge *et al.* showed that metadata features such as timestamps and addresses contaminate intrusion detectors [11]. Engelen et al., Liu et al. and Lanvin et al. documented processing and labelling errors in CIC-IDS datasets that changed reported results [12], [13], [24]. Guerra et al. argued that reliable labels for network traffic are harder to obtain than benchmark datasets imply [25]. Poor transfer between datasets, documented by Sarhan et al. and by Layeghy and Portmann for network flows [26], [27], is one visible consequence of such artefacts. Arp et al. and Apruzzese et al. listed sampling bias, spurious correlation and unrealistic evaluation settings among the most common problems in security machine learning [10], [28], [29].

2.3 Evaluating models on temporally structured data

When observations close together in time are more alike than observations far apart, random cross-validation leaks information from test to training through neighbouring samples. Roberts et al. showed this for ecological data and recommended blocked cross-validation [30]. Hewamalage et al. made the same point for forecasting [31], and Yang and Shami discussed it for IoT analytics under changing conditions [32]. Sensor telemetry is a textbook case: consecutive readings from a thermostat or weather station are nearly identical, so a random split almost guarantees that each test reading has a close twin in training.

Surveys of anomaly detection for IoT and sensor networks emphasise model families and seldom discuss evaluation design [33], [34]. The audit presented here is intended to fill that gap for one widely used benchmark.

III. DATA

The seven telemetry train–test files of the TON_IoT collection [35] were used as distributed. Each contains a date, a time, one to four sensor fields, a binary label and an attack type. Table 1 summarises the seven files. Every file holds 15,000 normal rows; the number and size of attack classes vary by device. Timestamps have one-second resolution.

Table 1: Summary of the TON_IoT telemetry train–test files. "Distinct states" counts unique combinations of sensor values. "Normal span" is the time between the first and last normal reading

Device	Rows	Classes	Sensor fields	Distinct states	Duplicates	Normal span
Fridge	39,944	7	temperature, temp_condition	151	2,834	39.5 h

Copyright © 2026 CJESR All Rights Reserved

Current Journal of Engineering and Science Research
Volume 3, Issue 11, pp 26-34, November 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.311004
Available Online at https://www.cjesr.com

GPS tracker	38,960	8	latitude, longitude	38,657	0	29.8 h
Garage door	39,587	8	door_state, sphone_signal	2*	20,221	39.4 h
Modbus	31,106	6	four function-code registers	8,865	13,314	3.8 min
Motion light	39,488	8	motion_status, light_status	2	13,658	39.5 h
Thermostat	32,774	7	current_temperature, thermostat_status	30,916	424	39.4 h
Weather	39,260	8	temperature, pressure, humidity	37,988	0	4.1 min

*After harmonising the encoding of sphone_signal (Section 5.4).

IV. AUDIT METHODOLOGY

4.1 Feature families

Three families of features were derived for each device.

Content features are the sensor values themselves. Numeric fields were used unchanged; categorical fields were one-hot encoded after harmonising inconsistent spellings.

Time features encode when a reading was taken: the day of the year and the second of the day, $s = 3600h + 60m + \sigma$, where h, m and σ are the hour, minute and second.

Acquisition features describe how readings arrived rather than what the readings said. For reading i with timestamp t_i in a device file sorted by time, the inter-reading gap is

$$g_i = t_i - t_{i-1} \tag{1}$$

and the burst count b_i is the number of readings sharing the timestamp t_i. Neither feature looks at a sensor value. In a deployed detector, sampling behaviour can be a legitimate signal. Here it is treated separately because, as Section 5.4 shows, it reflects how the capture was organised.

4.2 Splitting schemes

Two schemes were compared. *Random* splitting uses stratified five-fold cross-validation, as is common in the literature. *Blocked* splitting respects time within each class. Let the readings of class c be sorted by timestamp and divided into five contiguous blocks $B_{c,1}, \ldots, B_{c,5}$. Fold k then tests on

$$\mathcal{T}_k = \bigcup_c B_{c,k} \tag{2}$$

and trains on everything else. Every class is still present in every fold, but test readings are no longer interleaved with near-identical training readings. Blocking cannot remove the separation of classes in time, because that separation is built into the data. What it removes is the advantage of interpolating between neighbouring samples.

4.3 Information ceiling

For devices whose sensor values take few distinct states, the best possible accuracy from content alone can be computed exactly. Let $\mathcal{S}$ be the set of distinct content states and $n_{s,c}$ the number of rows in state s with class c. Any classifier that sees only the content must give the same answer for every row in a state, so its accuracy cannot exceed

$$A_{\max} = \frac{1}{N} \sum_{s \in \mathcal{S}} \max_c n_{s,c} \tag{3}$$

where N is the number of rows. $A_{\max}$ is a ceiling on content-only accuracy for any model.

4.4 Models and metrics

A decision tree, a random forest of 100 trees and a LightGBM ensemble of 200 trees (learning rate 0.05, 31 leaves) were trained on every combination of device, feature family and splitting scheme. Tree ensembles were chosen because tree-based models perform strongly on tabular data [36] and are the most common choice in the telemetry literature. The main metric is multiclass macro-F1,

$$F_1^{\text{macro}} = \frac{1}{K} \sum_{c=1}^{K} \frac{2P_c R_c}{P_c + R_c} \tag{4}$$

where P_c and R_c are the precision and recall of class c and K is the number of classes. Binary macro-F1 (normal against any attack) is also reported where it adds information. Results are averaged over the five folds.

V. RESULTS

5.1 Each class lives in its own time window

Figure 1 reconstructs the timeline of every device file. All normal readings were recorded between 31 March and 2 April 2019. The attacks followed three weeks later, between 23 and 29

Copyright © 2026 CJESR All Rights Reserved

Current Journal of Engineering and Science Research
Volume 3, Issue 11, pp 26-34, November 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.311004
Available Online at https://www.cjesr.com

April, and each attack type was run in its own interval: scanning on the evening of 23 April, injection on the morning of 25 April, DDoS that afternoon, password attacks on 26 April, XSS early on 27 April, ransomware early on 28 April, and backdoor later that day. The pattern is the same on every device. The only overlap occurs on the motion light, where scanning and password readings share part of the afternoon of 26 April.

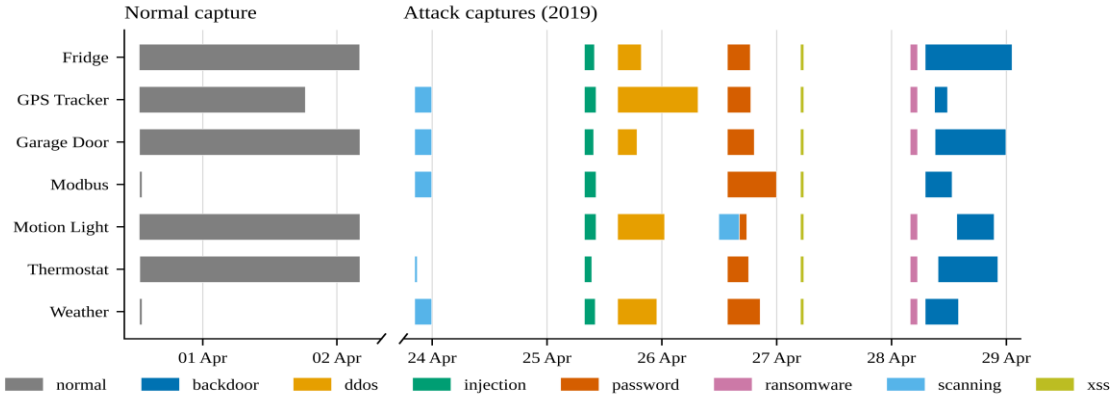


Figure 1: Recording windows for each class on each device. Normal data (left) were captured three weeks before any attack (right), and each attack type occupies its own interval

For Modbus and the weather station the problem is sharper. The 15,000 normal readings of these two devices span only 3.8 and 4.1 minutes respectively. Whatever a model learns about "normal" for these devices, it learns from four minutes of one afternoon.

5.2 The clock alone is almost enough

With nothing but day-of-year and second-of-day, the random forest reached a multiclass macro-F1 of 1.00 on six devices and 0.88 on the motion light under random splitting (Figure 2a). Blocked splitting barely changed this: scores of 0.91–0.99 on six devices and 0.78 on the motion light (Figure 2b). The motion light is the exception for a reason that confirms the rule, since it is the only device on which two classes overlap in time. Binary macro-F1 from time features was 1.00 for every device under both schemes.

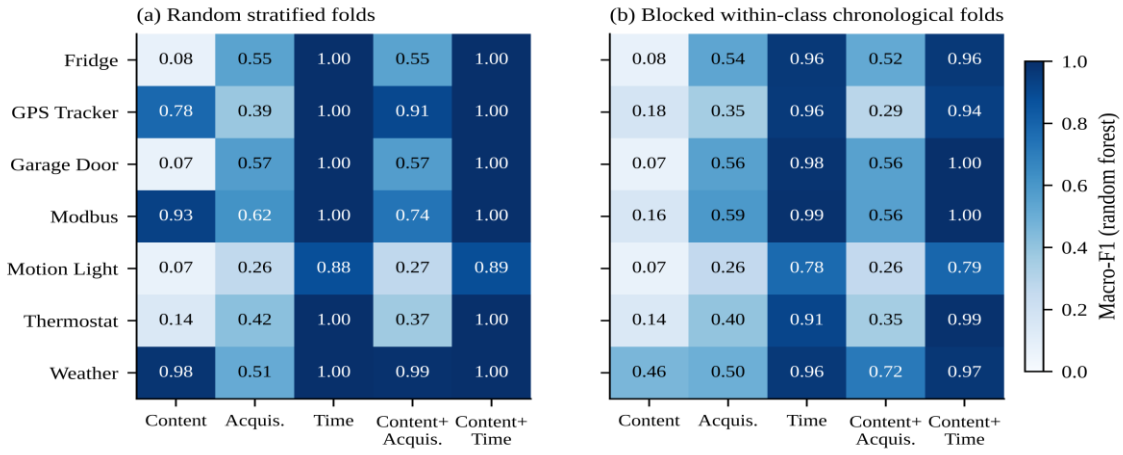


Figure 2: Random-forest macro-F1 by device and feature family under (a) random stratified folds and (b) blocked within-class chronological folds

Copyright © 2026 CJESR All Rights Reserved

Current Journal of Engineering and Science Research
Volume 3, Issue 11, pp 26-34, November 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.311004
Available Online at https://www.cjesr.com

Any model given the date and time fields, directly or through derived features such as hour or day, therefore has access to a near-perfect predictor of the label. Combining time with content produced the same scores as time alone. Once the clock is available, the sensor values add nothing.

5.3 Content scores depend on neighbouring readings

Sensor content alone gave a much more mixed picture (Figure 3). Under random splitting, macro-F1 ranged from 0.07 for the garage door and motion light to 0.93 for Modbus and 0.98 for the weather station. Under blocked splitting the high scores collapsed: Modbus fell from 0.93 to 0.16, the weather station from 0.98 to 0.46 and the GPS tracker from 0.78 to 0.18. Devices that scored poorly under random splitting scored the same under blocking, because those devices had nothing to lose.

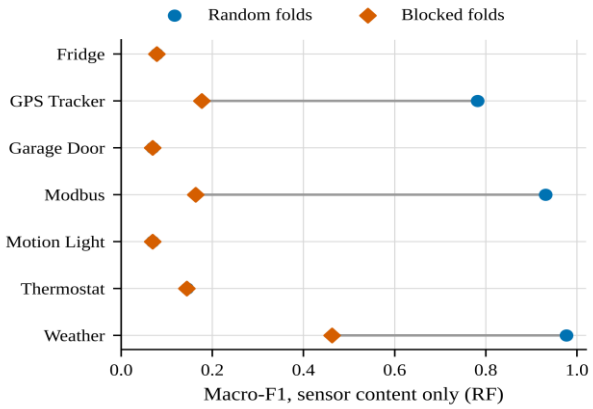


Figure 3: Content-only macro-F1 of the random forest under random and blocked folds

The explanation lies in Table 1. The weather station, GPS tracker and Modbus controller produce almost unique readings, 37,988 distinct states out of 39,260 rows in the weather case. Consecutive readings within one attack window are close to one another. Under random splitting, a test reading usually has a neighbour from the same window in training, and a tree ensemble can classify it by proximity. Under blocked splitting the nearest training readings come from a different stretch of time, and the apparent sensor signature disappears. The high random-split scores came from interpolating within a recording session, not from learning how an attack affects a temperature or a register value.

5.4 How the data were collected also reveals the label

Two acquisition artefacts are strong enough to classify the data without looking at any sensor value.

The first is sampling rate. Figure 4 shows the distribution of gaps between consecutive readings. On every device the median gap for normal data is zero seconds, meaning that several normal readings share each timestamp. During attack windows the median gap is one to two seconds. A classifier given only g_i and b_i achieved binary macro-F1 of 0.99 or higher for the garage door, Modbus controller and weather station, 0.95 for the fridge and 0.91 for the motion light. The normal captures appear to have been generated or replayed in bursts, while attack-period readings arrived at the device's ordinary pace.

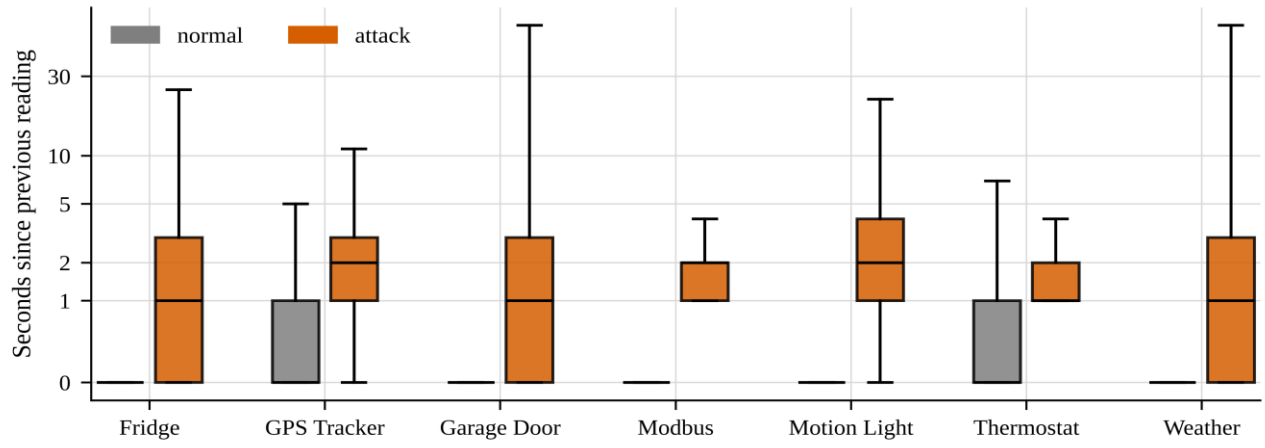


Figure 4: Distribution of the time since the previous reading for normal and attack data on each device (outliers omitted; logarithmic scale)

Copyright © 2026 CJESR All Rights Reserved

Current Journal of Engineering and Science Research
Volume 3, Issue 11, pp 26-34, November 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.311004
Available Online at https://www.cjesr.com

The second artefact is an encoding difference in the garage-door file. Its sphone_signal field is written as the strings true and false in every normal row and as 0 and 1 in every attack row (Table 2). A model that treats the raw field as a categorical variable, which is the default in most data-loading pipelines, receives a perfect binary label. The two encodings describe the same physical states: door_state is closed whenever the signal is 0 or false, and open whenever it is 1 or true.

Table 2: Joint distribution of the raw door_state and sphone_signal values by class in the garage-door file

Class	closed / "0"	open / "1"	closed / "false"	open / "true"
normal	0	0	13,478	1,522
backdoor	4,475	525	0	0
ddos	4,509	491	0	0
injection	4,496	504	0	0
password	4,510	490	0	0
ransomware	2,584	318	0	0
scanning	471	58	0	0
xss	1,052	104	0	0

A milder artefact appears in the GPS file, where normal latitudes are typically written with nine decimal places and attack-period latitudes with seven or eight. None of the other files showed systematic formatting differences between classes.

5.5 Three devices carry almost no information

After harmonising encodings, the garage door and motion light each have only two distinct content states, and the fridge has 151 (Table 1). For these devices every state occurs in several classes, so the ceiling in Equation (3) is low: $A_{\max} = 0.376$ for the fridge, 0.379 for the garage door and 0.380 for the motion light. These values are essentially the proportion of normal rows in each file. No classifier, however sophisticated, can beat the majority-class rate using these devices' sensor values alone. The measured content-only macro-F1 of 0.07–0.08 reflects exactly that.

The class proportions within each state confirm the point. In the motion-light file the light is off in 62% of normal readings and in 62–64% of the readings of every attack class. In the fridge file the "high" temperature condition accounts for 55–57% of every class. The attacks in these captures did not change what these devices reported.

The thermostat lies between the two groups. Its readings are nearly unique ($A_{\max} = 0.970$), yet content-only macro-F1 was 0.14 under both splitting schemes. Its values distinguish individual readings without distinguishing classes.

VI. DISCUSSION

6.1 What can and cannot be claimed from these files

Taken together, the results suggest that most published performance on the TON_IoT telemetry files can be explained without any model learning how attacks affect devices. If date and time are used in any form, the label is effectively given away. If date and time are removed but acquisition features remain, whether explicitly or through windowed statistics, the burst sampling of normal data gives the binary label away for most devices. If raw strings are used for the garage door, the encoding gives it away. If none of these is used and the split is random, high scores on the weather station, Modbus controller and GPS tracker come from interpolation between neighbouring readings. When all of these routes are closed, the best content-only macro-F1 across devices is 0.46.

This does not mean that telemetry is useless for security, or that studies using these files are wrong in every respect. It means that the files, as distributed, cannot support claims that a model detects attacks from sensor behaviour. The files can support methodological work on leakage, like this paper, and possibly work on attack-period anomalies within a single capture session, but not much more without additional care. The broader lesson echoes D'hooge et al. and Arp et al.: in security datasets, the collection process is part of the data, and models will learn it unless prevented [10], [11].

6.2 Recommendations for researchers

Researchers using these files should, at a minimum:

1. Drop the date and time fields and anything derived from these fields, including inter-reading gaps and window counts, unless timing is the explicit subject of study;
2. Harmonise categorical encodings before training, and check each field's raw values class by class;
3. Use blocked or time-aware cross-validation within classes rather than random splitting [30], [31];
4. Report the information ceiling for low-cardinality devices, so that readers can see how much any model could achieve;
5. Treat a near-perfect score on telemetry as a reason to look for an artefact, not as a result.

Copyright © 2026 CJESR All Rights Reserved

Current Journal of Engineering and Science Research
Volume 3, Issue 11, pp 26-34, November 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.311004
Available Online at https://www.cjesr.com

6.3 Recommendations for dataset maintainers

Three changes would make telemetry benchmarks of this kind far more useful. Normal and attack data should be recorded in interleaved sessions, ideally with normal operation continuing throughout each attack window, so that time cannot stand in for the label. Sampling should be identical across conditions, or explicitly documented if it differs. Every field should use one encoding throughout. These steps are cheap at collection time and nearly impossible to fix afterwards. Similar recommendations have been made for network datasets [12], [14], [25], and the evidence here suggests the same recommendations apply with greater force to device telemetry.

6.4 Limitations

The audit covers the curated train–test files rather than the full raw telemetry logs. The larger logs may contain normal data recorded during attack periods, which would allow a cleaner evaluation, and examining those logs is a natural next step. Only tree-based classifiers were used. Sequence models would face the same confounds, but the size of each effect could differ. The blocked scheme keeps classes in separate time windows and therefore cannot remove the time–class association itself; it only removes interpolation between neighbours. Finally, the inference that normal data were burst-sampled rests on timestamps with one-second resolution. Finer timestamps would show the sampling process more precisely.

VII. CONCLUSION

The TON_IoT telemetry files are widely used, and the scores reported on these files are high. This audit shows that, as distributed, the files allow the label to be recovered from the timestamp, from the reading rate, and on one device from a value encoding, without reference to what the sensors measured. Each attack type occupies its own window, normal data were captured weeks earlier and in bursts, and two devices hold only four minutes of normal operation. Once these routes are closed, content-only performance falls sharply, and three devices are shown to carry no class information at all. Future work on these files should state how each confound was handled, and future telemetry datasets should be collected so that the confounds never arise.

DATA AVAILABILITY

The TON_IoT datasets analysed in this study are publicly available from IEEE DataPort [35] (https://doi.org/10.21227/fesz-dm97) and from UNSW Canberra at https://research.unsw.edu.au/projects/toniot-datasets. The analysis scripts used in this study are available from the author on reasonable request.

CONFLICT OF INTEREST

The author declares no conflict of interest.

FUNDING

This research received no external funding.

REFERENCES

[1] A. Alsaedi, N. Moustafa, Z. Tari, A. Mahmood, and A. Anwar, “TON_IoT telemetry dataset: A new generation dataset of IoT and IIoT for data-driven intrusion detection systems,” *IEEE Access*, vol. 8, pp. 165130–165150, 2020, doi: 10.1109/ACCESS.2020.3022862.

[2] M. A. Umer, K. N. Junejo, M. T. Jilani, and A. P. Mathur, “Machine learning for intrusion detection in industrial control systems: Applications, challenges, and recommendations,” *International Journal of Critical Infrastructure Protection*, vol. 38, Art. no. 100516, 2022, doi: 10.1016/j.ijcip.2022.100516.

[3] A. M. Y. Koay, R. K. L. Ko, H. Hettema, and K. Radke, “Machine learning in industrial control system (ICS) security: Current landscape, opportunities and challenges,” *Journal of Intelligent Information Systems*, vol. 60, no. 2, pp. 377–405, 2023, doi: 10.1007/s10844-022-00753-1.

[4] N. Moustafa, “A new distributed architecture for evaluating AI-based security systems at the edge: Network TON_IoT datasets,” *Sustainable Cities and Society*, vol. 72, Art. no. 102994, 2021, doi: 10.1016/j.scs.2021.102994.

[5] N. Naz, M. A. Khan, S. A. Alsuhibany, M. Diyan, Z. Tan, M. A. Khan, and J. Ahmad, “Ensemble learning-based IDS for sensors telemetry data in IoT networks,” *Mathematical Biosciences and Engineering*, vol. 19, no. 10, pp. 10550–10580, 2022, doi: 10.3934/mbe.2022493.

[6] N. Naz, M. A. Khan, M. A. Khan, M. A. Khan, S. U. Jan, S. A. Shah, Arshad, Q. H. Abbasi, and J. Ahmad, “A comparison of ensemble learning for intrusion detection in telemetry data,” *in Advances on Intelligent Computing and Data Science, Lecture Notes on Data Engineering and Communications Technologies, vol. 179, Cham: Springer*, 2023, pp. 451–462, doi: 10.1007/978-3-031-

Copyright © 2026 CJESR All Rights Reserved

Current Journal of Engineering and Science Research
Volume 3, Issue 11, pp 26-34, November 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.311004
Available Online at https://www.cjesr.com

36258-3_40.

[7] S. Poonkuzhal, M. Shobana, and J. Jeyalakshmi, "A deep transfer learning approach for IoT/IIoTcyber attack detection using telemetry data," *Neural Network World*, vol. 33, no. 4, pp. 225–244, 2023, doi: 10.14311/NNW.2023.33.014.

[8] R. Geirhos, J.-H. Jacobsen, C. Michaelis, R. Zemel, W. Brendel, M. Bethge, and F. A. Wichmann, "Shortcut learning in deep neural networks," *Nature Machine Intelligence*, vol. 2, no. 11, pp. 665–673, 2020, doi: 10.1038/s42256-020-00257-z.

[9] S. Kapoor and A. Narayanan, "Leakage and the reproducibility crisis in machine-learning-based science," *Patterns*, vol. 4, no. 9, Art. no. 100804, 2023, doi: 10.1016/j.patter.2023.100804.

[10] D. Arp, E. Quiring, F. Pendlebury, A. Warnecke, F. Pierazzi, C. Wressnegger, L. Cavallaro, and K. Rieck, "Pitfalls in machine learning for computer security," *Communications of the ACM*, vol. 67, no. 11, pp. 104–112, 2024, doi: 10.1145/3643456.

[11] L. D'hooge, M. Verkerken, B. Volckaert, T. Wauters, and F. De Turck, "Establishing the contaminating effect of metadata feature inclusion in machine-learned network intrusion detection models," *in Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA 2022), LNCS, vol. 13358, Cham: Springer*, 2022, pp. 23–41, doi: 10.1007/978-3-031-09484-2_2.

[12] G. Engelen, V. Rimmer, and W. Joosen, "Troubleshooting an intrusion detection dataset: The CICIDS2017 case study*," in Proc. 2021 IEEE Security and Privacy Workshops (SPW)*, 2021, pp. 7–12, doi: 10.1109/SPW53761.2021.00009.

[13] M. Lanvin, P.-F. Gimenez, Y. Han, F. Majorczyk, L. Mé, and É. Totel, "Errors in the CICIDS2017 dataset and the significant differences in detection performances it makes," *in Risks and Security of Internet and Systems (CRiSIS 2022), LNCS*, vol. 13857, *Cham: Springer*, 2023, pp. 18–33, doi: 10.1007/978-3-031-31108-6_2.

[14] T. M. Booij, I. Chiscop, E. Meeuwissen, N. Moustafa, and F. T. H. den Hartog, "ToN_IoT: The role of heterogeneity and the need for standardization of features and attack types in IoT network intrusion data sets," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 485–496, 2022, doi: 10.1109/JIOT.2021.3085194.

[15] S. Haque, F. El-Moussa, N. Komninos, and M. Rajarajan, "A systematic review of data-driven attack detection trends in IoT," *Sensors*, vol. 23, no. 16, Art. no. 7191, 2023, doi: 10.3390/s23167191.

[16] A. Heidari and M. A. Jabraeil Jamali, "Internet of Things intrusion detection systems: A comprehensive review and future directions," *Cluster Computing*, vol. 26, no. 6, pp. 3753–3780, 2023, doi: 10.1007/s10586-022-03776-z.

[17] E. Gyamfi and A. Jurcut, "Intrusion detection in Internet of Things systems: A review on design approaches leveraging multi-access edge computing, machine learning, and datasets," *Sensors*, vol. 22, no. 10, Art. no. 3744, 2022, doi: 10.3390/s22103744.

[18] M. A. Ferrag, O. Friha, D. Hamouda, L. Maglaras, and H. Janicke, "Edge-IIoTset: A new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning," *IEEE Access*, vol. 10, pp. 40281–40306, 2022, doi: 10.1109/ACCESS.2022.3165809.

[19] M. Al-Hawawreh, E. Sitnikova, and N. Aboutorab, "X-IIoTID: A connectivity-agnostic and device-agnostic intrusion data set for industrial Internet of Things," *IEEE Internet of Things Journal*, vol. 9, no. 5, pp. 3962–3977, 2022, doi: 10.1109/JIOT.2021.3102056.

[20] E. C. P. Neto, S. Dadkhah, R. Ferreira, A. Zohourian, R. Lu, and A. A. Ghorbani, "CICIoT2023: A real-time dataset and benchmark for large-scale attacks in IoT environment," *Sensors*, vol. 23, no. 13, Art. no. 5941, 2023, doi: 10.3390/s23135941.

[21] S. Kaufman, S. Rosset, C. Perlich, and O. Stitelman, "Leakage in data mining: Formulation, detection, and avoidance," *ACM Transactions on Knowledge Discovery from Data*, vol. 6, no. 4, pp. 1–21, 2012, doi: 10.1145/2382577.2382579.

[22] M. Rosenblatt, L. Tejavibulya, R. Jiang, S. Noble, and D. Scheinost, "Data leakage inflates prediction performance in connectome-based machine learning models," *Nature Communications*, vol. 15, no. 1, Art. no. 1829, 2024, doi: 10.1038/s41467-024-46150-w.

[23] S. Whalen, J. Schreiber, W. S. Noble, and K. S. Pollard, "Navigating the pitfalls of applying machine learning in genomics," *Nature Reviews Genetics*, vol. 23, no. 3, pp. 169–181, 2022, doi: 10.1038/s41576-021-00434-9.

[24] L. Liu, G. Engelen, T. Lynar, D. Essam, and W. Joosen, "Error prevalence in NIDS datasets: A case study on CIC-IDS-2017 and CSE-CIC-IDS-2018," *in Proc. 2022 IEEE Conference on Communications and Network Security (CNS)*, 2022, pp. 254–262, doi: 10.1109/CNS56114.2022.9947235.

[25] J. L. Guerra, C. Catania, and E. Veas, "Datasets are not

Copyright © 2026 CJESR All Rights Reserved

Current Journal of Engineering and Science Research
Volume 3, Issue 11, pp 26-34, November 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.311004
Available Online at https://www.cjesr.com

enough: Challenges in labeling network traffic," *Computers & Security*, vol. 120, Art. no. 102810, 2022, doi: 10.1016/j.cose.2022.102810.

[26] M. Sarhan, S. Layeghy, and M. Portmann, "Evaluating standard feature sets towards increased generalisability and explainability of ML-based network intrusion detection," *Big Data Research*, vol. 30, Art. no. 100359, 2022, doi: 10.1016/j.bdr.2022.100359.

[27] S. Layeghy and M. Portmann, "Explainable cross-domain evaluation of ML-based network intrusion detection systems," *Computers and Electrical Engineering*, vol. 108, Art. no. 108692, 2023, doi: 10.1016/j.compeleceng.2023.108692.

[28] G. Apruzzese, P. Laskov, and J. Schneider, "SoK: Pragmatic assessment of machine learning for network intrusion detection," in *Proc. 2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P)*, 2023, pp. 592–614, doi: 10.1109/EuroSP57164.2023.00042.

[29] G. Apruzzese, P. Laskov, E. Montes de Oca, W. Mallouli, L. Brdalo Rapa, A. V. Grammatopoulos, and F. Di Franco, "The role of machine learning in cybersecurity," *Digital Threats: Research and Practice*, vol. 4, no. 1, pp. 1–38, 2023, doi: 10.1145/3545574.

[30] D. R. Roberts, V. Bahn, S. Ciuti, M. S. Boyce, J. Elith, G. Guillera-Arroita, et al., "Cross-validation strategies for data with temporal, spatial, hierarchical, or phylogenetic structure," *Ecography*, vol. 40, no. 8, pp. 913–929, 2017, doi: 10.1111/ecog.02881.

[31] H. Hewamalage, K. Ackermann, and C. Bergmeir, "Forecast evaluation for data scientists: Common pitfalls and best practices," *Data Mining and Knowledge Discovery*, vol. 37, no. 2, pp. 788–832, 2023, doi: 10.1007/s10618-022-00894-5.

[32] L. Yang and A. Shami, "IoT data analytics in dynamic environments: From an automated machine learning perspective," *Engineering Applications of Artificial Intelligence*, vol. 116, Art. no. 105366, 2022, doi: 10.1016/j.engappai.2022.105366.

[33] K. DeMedeiros, A. Hendawi, and M. Alvarez, "A survey of AI-based anomaly detection in IoT and sensor networks," *Sensors*, vol. 23, no. 3, Art. no. 1352, 2023, doi: 10.3390/s23031352.

[34] A. Chatterjee and B. S. Ahmed, "IoT anomaly detection methods and applications: A survey," Internet of Things, vol. 19, Art. no. 100568, 2022, doi: 10.1016/j.iot.2022.100568.

[35] N. Moustafa, "ToN_IoT datasets," *IEEE Dataport*, Oct. 16, 2019. [Online]. Available: https://ieee-dataport.org/documents/toniot-datasets, doi: 10.21227/fesz-dm97.

[36] R. Shwartz-Ziv and A. Armon, "Tabular data: Deep learning is not all you need," *Information Fusion*, vol. 81, pp. 84–90, 2022, doi: 10.1016/j.inffus.2021.11.011.

AUTHOR'S BIOGRAPHY



Engr. Dr. Reginald Ogu is a lecturer and researcher in the Department of Computer Engineering, Federal University of Technology, Owerri (FUTO), Nigeria. Research interests include machine learning for cybersecurity, Internet of Things security, ICT capacity building and digital inclusion. Dr. Ogu is active in the IEEE Computer Society and serves as advisor to the IEEE IAS FUTO Student Branch Chapter.

Citation of this Article:

Reginald Ekene Ogu. (2026). An Audit of Temporal and Acquisition Confounds in the TON_IoT Telemetry Datasets. *Current Journal of Engineering and Science Research*. 3(11), 26-34. Article DOI: https://doi.org/10.47001/CJESR/2026.311004

***** End of the Article *****

Copyright © 2026 CJESR All Rights Reserved