IoT-CAD: Context-Aware Adaptive Anomaly Detection in IoT Systems Through Sensor Association

Rozhin Yasaei, Felix Hernandez, Mohammad Abdullah Al Faruque
Department of Electrical Engineering and Computer Science
University of California, Irvine, California, USA
{ryasaei,felixh1,alfaruqu}@uci.edu

ABSTRACT

The deployment of Internet of Things (IoT) devices in cyber-physical applications has introduced a new set of vulnerabilities. The new security and reliability challenges require a holistic solution due to the cross-domain, cross-layer, and interdisciplinary nature of IoT systems. However, the majority of works presented in the literature primarily focus on the cyber aspect, including the network and application layers, and the physical layer is often overlooked.

In this paper, we utilize IoT sensors that capture the physical properties of the system to ensure the integrity of IoT sensors data and identify anomalous incidents in the environment. We propose an adaptive context-aware anomaly detection method that is optimized to run on a fog computing platform. In this approach, we devise a novel sensor association algorithm that generates fingerprints of sensors, clusters them, and extracts the context of the system. Based on the contextual information, our predictor model, which comprises an Long-Short Term Memory (LSTM) neural network and Gaussian estimator, detects anomalies, and a consensus algorithm identifies the source of the anomaly. Furthermore, our model updates itself to adapt to the variation in the environment and system. The results demonstrate that our model detects the anomaly with 92.0% precision in 532ms, which meets the real-time constraint of the system under test.

CCS CONCEPTS

• **Computer systems organization** → **Embedded and cyber-physical systems**.

KEYWORDS

Internet of Things, context, sensor association, anomaly detection, recurrent neural networks, LSTM encoder-decoder

1 INTRODUCTION

Over the last decade, IoT has grabbed substantial attention due to advancements in computation and communication, and it is utilized in many applications such as smart home, automotive, and medical aid. The rapid growth of IoT has raised concerns about the security and reliability of these systems. There are a tremendous amount of work in the literature that focuses on various aspects of IoT systems such as communication network [24, 38], hardware security [5, 15, 21, 22] or software security [3, 34, 40, 41]. However, the physical layer of IoT as a cyber-physical system (CPS) is overlooked. To ensure the security of CPS systems, in addition to a bottom-up security attitude, a holistic approach is required [8–10, 12].

ICCAD '20, November 2–5, 2020, Virtual Event, USA
© 2020 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-8026-3/20/11…$15.00
https://doi.org/10.1145/3400302.3415672

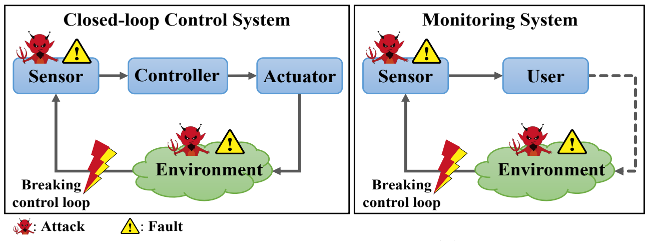


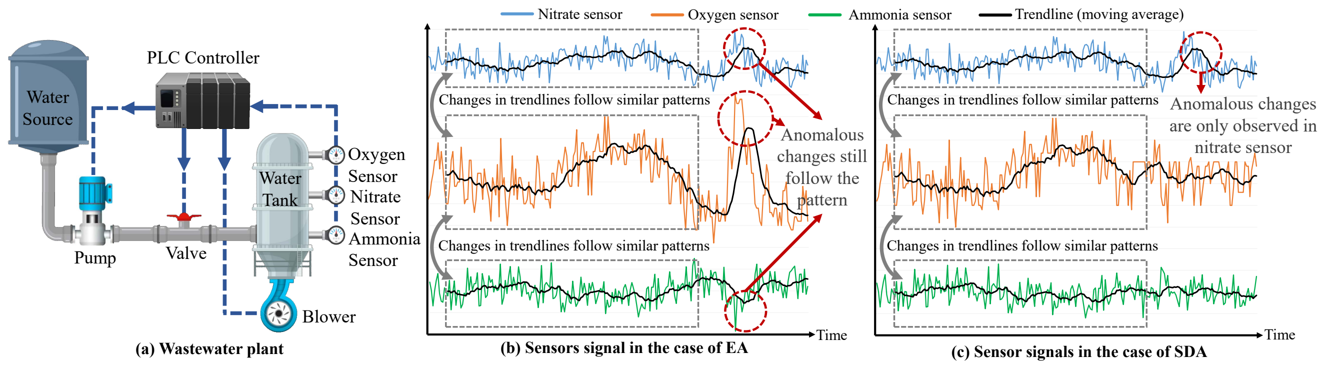
Figure 1: Two categories of IoT systems; (a) Closed-loop control system, and (b) Monitoring system.

The ultimate goal of an IoT system is to control the environment and maintain it in the desired state. In order to explain the important role of sensors in fulfilling this goal, we categorize IoT systems under two categories, as depicted in Figure 1: (i) a *closed-loop control* system, and (ii) a *monitoring system*. On the one hand, a *closed-loop control system* consists of three major components: (i) sensors; (ii) controller; and (iii) actuators (see Figure 1(a)). The sensors monitor the system and send the status to the controller, which processes the sensor readings, decides how to react, and sends the control signals to the actuators to maintain the state of system and environment.

On the other hand, *monitoring systems* mainly contain sensors that measure numerous parameters in the system and provide the user with information to take proper action (see Figure 1(b)). Although a *monitoring system* cannot directly manipulate the environment, it informs a supervising user of events that happen in the system, and the user controls the system manually. Thus, a monitoring system is eventually a part of a control loop.

In both categories, sensors are an essential component of the control loop since sensor measurements determine the action that is needed to maintain the system in the desired state. Malfunction or manipulation of a sensor can break the control loop [4], and consequently, disrupt the services offered by the IoT system. Fault in a sensor device leads to the appearance of anomalous values in its readings, whereas not all anomalies in sensor measurements indicate sensor breakage because an unexpected event in the environment may cause an anomaly as well. Observing the possible anomalies in an IoT system, we present a classification of anomalies which facilitates identification of the anomaly's source:

- **Environmental Anomaly (EA):** The environment is the area that surrounds the sensor, and the sensor measures its physical properties. Any anomaly in the environment affects the measurements of the sensor and disrupts it. An EA may occur as a result of malicious activities or unexpected incidents in the environment.



Water Source
PLC Controller
Water Tank
Oxygen Sensor
Nitrate Sensor
Ammonia Sensor
Pump
Valve
Blower
(a) Wastewater plant
Nitrate sensor
Oxygen sensor
Ammonia sensor
Trendline (moving average)
Changes in trendlines follow similar patterns
Anomalous changes still follow the pattern
Changes in trendlines follow similar patterns
Time
(b) Sensors signal in the case of EA
Changes in trendlines follow similar patterns
Anomalous changes are only observed in nitrate sensor
Changes in trendlines follow similar patterns
Time
(c) Sensor signals in the case of SDA

Figure 2: (a) Schema of wastewater plant, and synthetic sensors' signals in the (b) first scenario (EA), (c) second scenario (SDA).

- **Sensing Device Anomaly (SDA):** When the operation of a sensor is corrupted, its measurements do not follow the same pattern, and an SDA is observed. This corruption occurs because of either security or reliability issues. For instance, [1, 45] discuss some attacks on the physical layer.

Current anomaly detection methods model the normal behavior of a device [7, 13, 31, 33] and label any deviation from expected behavior as an anomaly. Most of the works concentrate on anomaly detection in the network layer of IoT systems [20]. In spite of reasonable performance in network intrusion detection, these methods have a high rate of false alarms when used with sensor signals. They misinterpret the environmental variation in the sensors measurements as an SDA and disregard the potential information encoded in the relation between the system and the physical world, known as the **context** of the system (refer to Sec. 3.1 for the definition of context). Conventionally, context-aware methods are applied to a variety of applications [2], and recently, these methods are used to secure the authentication of co-located devices [16, 35, 36, 42].

In this work, we propose an adaptive data-driven model for unsupervised anomaly detection in IoT systems based on the sensor measurements. The model monitors the system to detect anomalies, identifies the type of anomaly (SDA or EA), and locates them. To this end, we develop an algorithm to extract the patterns in sensor signals and generate the context of system. Then, we associate the sensors from different modalities based on the context and cluster the sensors with similar behavior. We develop our customized Recurrent Neural Network (RNN), followed by a consensus algorithm to detect and localize anomalies. The consensus algorithm checks the consistency between sensors in each cluster and determines the type of anomaly. An IoT system has a dynamic structure that is open to changes, such as adding new nodes, removing the existing ones, or updating the framework and protocols. In order to address the variation in IoT systems over time, our model is designed to be adaptive and update itself.

1.1 Motivational Example

As a real-world IoT system, we study the environmental training center wastewater plant in Riccione [14]. The primary purposes of wastewater treatment is the elimination of nitrate. Nitrate contamination is a severe environmental problem because it can exhibit toxicity toward aquatic life, present a public health hazard, and affect the suitability of wastewater. In the treatment process, the wastewater is pumped to the tanks, which are equipped with sensors to monitor the concentration of oxygen, ammonia, and nitrate in the water. The actuators, such as blowers and valves, are controlled by a Programmable Logic Controller to adjust the level of chemicals (Figure 2(a)). Given the importance of the nitrate level, anomaly detection is applied to detect abnormal changes. Consider two scenarios with anomalous rise in nitrate level; In the first scenario, environmental changes alter the water temperature, which affects the chemical reactions in the water tank (Figure 2(b), an example of EA). In the second scenario, the nitrate sensor is broken or manipulated by an attacker. (Figure 2(c), an example of SDA). The current anomaly detection methods rely solely on nitrate sensor data, whereas the validity of its data is questionable. Thus, they can not find the source of anomaly and discriminate EA and SDA.

A recent study [14] analyzes the sensors of this wastewater plant and reveals the correlation between ammonia, oxygen, and nitrate sensor data. More specifically, when the rise in oxygen density reaches a certain threshold, the ammonia concentration decreases, and the nitrate concentration increases. Further investigation reveals the scientific rationale for this correlation; oxygen triggers the chemical reaction, which affects the ammonia and nitrate concentration. By considering this relationship, it is possible to validate sensor signals. In the first scenario, the incident affects all sensors. Despite irregularities in the sensor signals, they are consistent with each other. Thus, we can conclude that the integrity of the sensors' data is not compromised. In the second scenario, the anomaly in the nitrate sensor data is inconsistent with the patterns of other sensor signals. It indicates that the cause of the abnormality is fault or attack. This type of relationship between sensors in not limited to this wastewater plant and it is observed in many IoT systems due to the availability of many heterogeneous sensors.

1.2 Threat Model

The proposed methodology aims to detect SDA and EA, which occur due to an unexpected incident in the environment, reliability issue, or security breakage. Accidental damage, degradation, and defects are examples of plausible reliability problems that cause unintended device malfunctions. In contrast, the security breakage scenario involves an attacker who intentionally exploits the vulnerabilities in the system. In this threat model, the adversary has access to the sensor device and fiddles with it to inject fault, alter functionality, or deny its service. As another possible scenario, the attacker can control the communication channel and send faulty signal to the controller as sensor measurements. The model can detect anomalies in a standalone sensor, but to distinguish between SDA and EA in a sensor, it should be associated with at least two other sensors. To deceive this method, the attacker should be able to discover how sensors are clustered, learn the correlations and patterns in the sensors' signals, and manipulate them in a way that imitates the same correlation as before. It means that in addition to sensors, the attacker should have full access to the clustering layout of sensors and the trained anomaly detection model. It is assumed that the attacker does not have these privileges.

1.3 Research Challenges

Anomaly detection in the IoT sensors is challenging due to the following reasons [11]:

- The IoT data are multi-variant time-series data that are collected from a heterogeneous network of sensors with different modalities, data dimensions, sampling rates, specifications, and locations.
- Low cost and resource-constrained sensors are usually sensitive to noise, and deployment of them in IoT systems affects the quality of data.
- Due to lack of prior knowledge about possible anomalies and scarcity of anomalous observations, there is not enough labeled anomalous data available, and conventional supervised machine learning technique are not applicable.
- IoT systems have dynamic characteristics that may be altered over time because of environmental changes, human interaction, mobility of devices, and updating firmware or software. Consequently, a static model fails to imitate the system in the long-term.

1.4 Our Contributions

To the best of our knowledge, this is the first context-aware anomaly detection method for IoT systems. Our novel contributions to address the aforementioned challenges are summarized below:

- **Context-aware sensor association algorithm**: We develop a multi-modality clustering method to associate sensors that experience similar contextual variation.
- **Consensus-based strategy for unsupervised anomaly detection**: We design a methodology to pinpoint the anomalies without reliance on prior knowledge about possible anomalies.
- **Adaptive data-driven model**: Our proposed anomaly detection model is periodically updated at run-time to adapt itself to new states caused by variations in the system.

2 RELATED WORKS

General anomaly detection algorithms can be classified into the following main categories [14]:

Statistical or Probabilistic Methods: These methods create a statistical or probabilistic model based on history data, which represents normal behavior [17, 39]. Upcoming observation is then compared with this model, and it is marked as an anomaly if it is statistically unlikely, or the probability of such observation is low.

Proximity Methods: These methods compute distances between data points to differentiate between anomalous and normal data. Two well-known techniques that fall in this category are the *Local Outlier Factor* [6] and *clustering* [18] methods.

Predictive Methods: In these methods, the anomaly detection problem is converted to obtaining an accurate sequence prediction algorithm that captures the recent and long-term trends in data

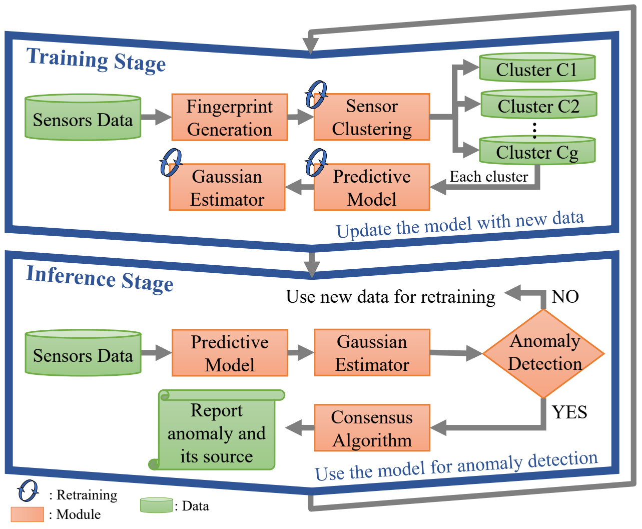


Figure 3: The architecture of our methodology in the training and inference stage.

sequences and reproduces them to predict future measurements. Afterward, the predictions are compared with the new observations to spot deviations from expected normal behavior. Recurrent Neural Networks (RNN) are capable of capturing the relationship between measurements over time because the feedback loops in the hidden layer of RNN can imitate memory.

Long-Short Term Memory (LSTM) layer was introduced in 1997 by [19] to overcome the shortcomings of RNN. It has gained a lot of attention lately because of its high accuracy in sequence prediction [7, 31, 33]. Conv-LSTM encoder-decoder is one of the neural network architectures that is used in the literature to enhance sequence prediction performance [23, 25, 27, 43, 44]. It contains convolutional layers to extract the essential features of input sequences and LSTM layers to perform the sequence prediction based on the features. Then, the anomaly is identified based on the reconstruction error of the model. LSTM-LSTM encoder-decoder [32, 37, 46] is another popular architecture which follows a similar strategy but it utilize LSTM layers instead of convolutional layers for feature extraction.

Our methodology inherent the advantages of both probabilistic and predictive methods. We implement and compare the Conv-LSTM and LSTM-LSTM encoder-decoder as our predictive models. Then, the reconstruction error, derived from the difference between real and predicted values, is modeled by a Multivariate Gaussian Estimators to detect the anomaly.

3 ANOMALY DETECTION METHODOLOGY

Our proposed methodology (see Figure 3) detects SDA and EA in an IoT system to ensure sensing devices operate as they are expected.

3.1 Context Generation

The context of a system is defined as an abstraction formed by extracting features from system circumstances and individual element constructs[26]. It describes the condition in which the system is operating and affects the outcome of the system. The first step for obtaining our context-aware data-driven model is to generate the context of the system by encoding its physical properties. Understanding and transforming this information such that it can be mathematically described is called **context generation**. Following

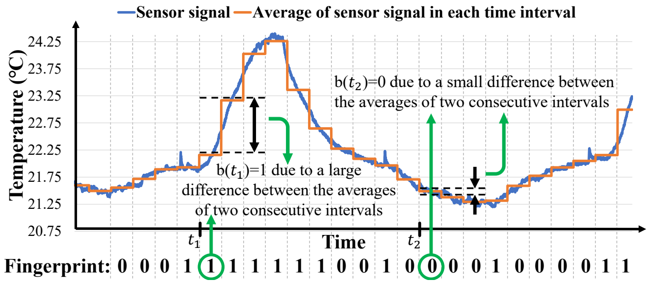


Figure 4: Extracting the fingerprint of a temperature sensor.

the strategy presented by Sadeghi et al. in [35], we convert all sensor signals to binary fingerprints regardless of their modality. The procedure of fingerprint generation has the following steps:

Step1: Each sensor continuously monitors the environment by taking a measurement each z seconds. The value z depends on the sampling rate of sensor and may vary for different sensors. In a time window of q seconds from timestamp t, the sensor records $v = [q/z]$ measurements and forms a snapshot vector $S_t = (s^t, s^{t+z}, \ldots, s^{t+z(v-1)})$.

Step2: The ϵ_{rel} is a pre-defined threshold that controls the amount of variation that is said to conform a change. The values obtained in a snapshot are averaged and the variation bit $b(t)$ is calculated as follows:

$$\overline{S}_t = \frac{1}{v}\sum_{s \in S_t} s, \; b(t) = \begin{cases} 1, & \text{if } \left|\frac{\overline{S}_{t+z} - \overline{S}_t}{\overline{S}_t}\right| > \epsilon_{rel} \\ 0, & \text{o.w.} \end{cases}$$

Step3: Finally, a sequence of $k+1$ consecutive snapshots $seq(t, t+kz) = (\overline{S}_t, \overline{S}_{t+z}, \ldots, \overline{S}_{t+kz})$, has an associated fingerprint $F(seq(t, t+kz)) = (b(t), b(t+z), \ldots, b(t+(k-1)z))$. The fingerprints of all the sensors with different sampling rates have the same length because each snapshot is the average of sensor measurements in a particular time interval. Figure 4 illustrates the process of generating the fingerprint of a temperature sensor.

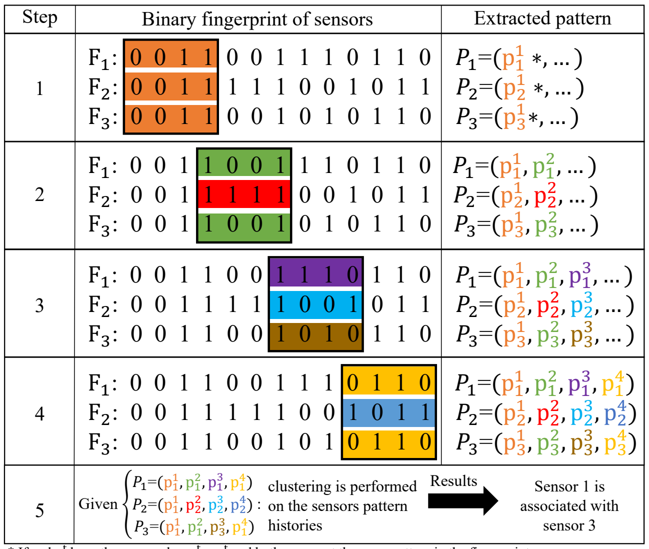
3.2 Sensor Association

Although each sensor's measurements differ based on its modality and physical location, the sensors that are affected by the same event follow similar patterns in their fingerprints. Based on this observation, we develop a sensor association algorithm that comprises two primary steps: I) pattern extraction and II) sensor clustering.

In the first step, we split each fingerprint into smaller sub-sequences, and cluster the sub-sequences of different sensors that have a similar binary pattern. For simplicity, assume that $F_i = F(seq(t, t+zk)_i)$ represents the fingerprint of the sensor i, which is split into d smaller sub-sequences ${f_i}^j$ as follows:

$$F_i \longrightarrow ({f_i}^1, {f_i}^2, \ldots, {f_i}^d), \; d = \frac{k-o}{l-o}$$

where l and o are the hyper parameters that determine the sub-sequences length and their overlap accordingly. Afterward, our clustering algorithm is performed on the sub-sequences of index j ($j \in [1, d]$) of all sensors $({F_1}^j, {F_2}^j, \ldots, {F_n}^j)$ to group the ones with similar binary patterns. Hence, it assign a pattern number $p_i^j \in \{0, 1, ..., p_{max}^j\}$ to ${f_i}^j$. Next, the clustering is repeated for index $j+1$ and after d iterations, all sub-sequences are clustered. Notice that



Step	Binary fingerprint of sensors	Extracted pattern
1	F_1: 0 0 1 1 0 0 1 1 1 0 1 1 0	P_1=(p_1^1 *, ...)
	F_2: 0 0 1 1 1 1 1 0 0 1 0 1 1	P_2=(p_2^1 *, ...)
	F_3: 0 0 1 1 0 0 1 0 1 0 1 1 0	P_3=(p_3^1*, ...)
2	F_1: 0 0 1 1 0 0 1 1 1 0 1 1 0	P_1=(p_1^1, p_1^2, ...)
	F_2: 0 0 1 1 1 1 1 0 0 1 0 1 1	P_2=(p_2^1, p_2^2, ...)
	F_3: 0 0 1 1 0 0 1 0 1 0 1 1 0	P_3=(p_3^1, p_3^2, ...)
3	F_1: 0 0 1 1 0 0 1 1 1 0 1 1 0	P_1=(p_1^1, p_1^2, p_1^3, ...)
	F_2: 0 0 1 1 1 1 1 0 0 1 0 1 1	P_2=(p_2^1, p_2^2, p_2^3, ...)
	F_3: 0 0 1 1 0 0 1 0 1 0 1 1 0	P_3=(p_3^1, p_3^2, p_3^3, ...)
4	F_1: 0 0 1 1 0 0 1 1 1 0 1 1 0	P_1=(p_1^1, p_1^2, p_1^3, p_1^4)
	F_2: 0 0 1 1 1 1 1 0 0 1 0 1 1	P_2=(p_2^1, p_2^2, p_2^3, p_2^4)
	F_3: 0 0 1 1 0 0 1 0 1 0 1 1 0	P_3=(p_3^1, p_3^2, p_3^3, p_3^4)
5	Given $\begin{cases} P_1\text{=}(p_1^1, p_1^2, p_1^3, p_1^4) \\ P_2\text{=}(p_1^1, p_2^2, p_2^3, p_2^4) \\ P_3\text{=}(p_1^1, p_1^2, p_3^3, p_1^4) \end{cases}$: clustering is performed on the sensors pattern histories	Results ➡ Sensor 1 is associated with sensor 3

* If and p_i^t have the same color, $p_i^t = p_j^t$ and both represent the same pattern in the fingerprint.
** In this example, o=1, l=4, and d=4.

Figure 5: The procedure of extracting the patterns in sensor signals and clustering them.

p_{max}^j which represents the number of clusters for index j may vary for different index values. Eventually, for each sensor the pattern numbers form a *pattern history vector* $P_i = ({p_i}^1, {p_i}^2, \ldots, {p_i}^d)$.

In the second step, one final clustering is performed on the given set of sensors pattern history P_i to determine the sensors cluster layout $C = c_1, c_2, ...c_g$ where c_i represents a clusters and g is the number of sensor clusters. The sensors with similar contextual variations exhibit the same patterns in many sub-sequences and we cluster them together. An example of the sensor association procedure is demonstrated in Figure 5. In this example, the final clustering group the first and third sensors are grouped together.

All the mentioned clustering processes are done using our customized clustering algorithm which minimizes the distance between data points in the same cluster, the *intra-cluster distance (IC)*, and maximizes the distance among data point of one cluster from other cluster data points, the *inter-cluster distance (OC)*. The distance matrics are defined as follows:

$$IC = \overline{IC_i}, \; IC_i = \frac{1}{|c_i|} \sum_{m,k \in c_i} Hamming(P_m, P_k)$$

$$OC = \overline{OC_{i,j}}, \; OC_{i,j} = \min_{m \in c_i, k \in c_j} \{Hamming(P_m, P_k)\}$$

Where c_i and P_m represents a cluster and the pattern history of sensor m respectively. Algorithm 1 is a Pseudo-code that elaborates on our clustering algorithm. Our clustering has the following properties:

- It can be applied to data with string type because the distance matrics are based on the Hamming distance function, which calculates the number of non-matching bits.
- The number of clusters is automatically tuned. Initially, clustering is performed with an upper bound of the number of clusters. Afterward, the algorithm automatically removes the nodes which are not close to any cluster and eliminates clusters with two nodes to reach optimum value for the number of clusters.

After sensor association, we evaluate the system to ensure that there is no standalone sensor which is not clustered. A standalone sensor is vulnerable because it is not related to any group of sensors that can verify its proper operation. In this case, the anomaly detection still can be applied to the independent sensor individually, but the SDA and EA are indistinguishable. The user is warned about this vulnerability in sensors and can resolve the issue by adding more sensors to the system.

Algorithm 1: Customized clustering algorithm for extracting patterns in sensor fingerprints and sensor association.

```
Input: Fingerprints: $F \in \mathbb{R}^{n \times k}$, number of sensors: $n$,
       sub-sequence length: $l$, overlap: $o$
Output: Cluster layout $C = \{c_1, c_2, \dots, c_g\}$
Initialize $d = \frac{k-o}{l-o}$
Initialize $p_{max}$ = max # of patterns in sub-sequence
Initialize $iter_{max}$ = max # of iterations
Initialize the center of clusters randomly
foreach $j \in \{1, 2, \dots, d\}$ do
    foreach $i \in \{1, 2, \dots, n\}$ do
        Split fingerprint $F_i$ to obtain sub-sequences ${f_i}^j$;
    Clustering:
    foreach $iter \in \{1, 2, \dots, iter_{max}\}$ do
        foreach $i \in \{1, 2, \dots, n\}$ do
            $p_i^j = Argmin_{x \in C} Hamming(f_i^j, center(x))$;
        foreach $x \in \{1, 2, \dots, p_{max}\}$ do
            $center(x) = mean(\{f_i^j | p_i^j = x\})$;
        if no changes in $center(x)$ then
            break;
    foreach $x \in \{1, 2, \dots, p_{max}\}$ do
        Calculate inter-cluster ($OC$) metrics;
        if $Hamming(F_i^j, center(p_i^j)) > OC$ then
            Remove $F_i^j$ from cluster $p_i^j$;
            Add $F_i^j$ in unclustered nodes;
            Update $p_{max}^j$;
        if $|c_i| < 3$ then
            Remove cluster $x$;
            Update $p_{max}^j$;
        Add clusters to pattern histories $P_i$;
Perform the clustering again on pattern histories $P_i$,
  $i \in [1, n]$ to associate sensors;
return Sensors Cluster layout $C = \{c_1, \dots, c_g\}$
```

3.3 Predictive Model

The next module of our methodology is the predictive model that predicts the future measurements of sensors according to the clustering layout and history of measurements. We construct a Recurrent Neural Networks (RNN) for each cluster of sensors as the predictive model. As it is depicted in 6, our RNN comprises LSTM encoder-decoder and dense layers, which encode the features of input sequences of length l_i and predict the future sequences of length l_o based on the encoded features. Sequences of data are derived from the input time-series signals using the sliding window technique. Afterward, the sequences are scaled through a **Min-Max Scaler** before being treated by the encoder because input signals come from

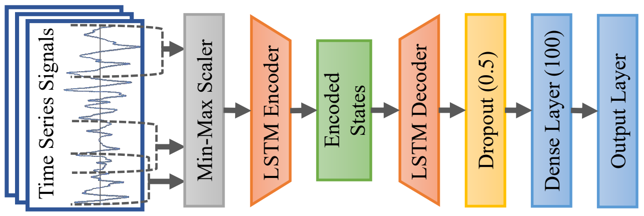


Figure 6: The architecture of RNN used as predictive model.

multi-modality sensors with different signal ranges. Eventually, we have a set of predictive models $DT = \{M_1, M_2, \dots, M_g\}$ where g is the number of clusters in the system and M_i represents the model for cluster c_i. Given cluster c_i that contains n_i nodes, the model M_i takes as input a matrix $X_i \in \mathbb{R}^{l_i \times n_i}$ to predict another matrix $Y_i \in \mathbb{R}^{n_i \times l_o}$. On top of predictive models, **Multivariate Gaussian Estimators** are trained to learn the probability of finding a particular error vector. This probability is used to ascertain whether the errors between predictions and real measurements correspond to the system's normal behavior, or an anomaly has occurred. A multivariate Gaussian distributor $G_i = \mathcal{N}(\mu_i, \sigma_i)$ is fitted on the reconstruction error matrix E_i, which is the difference between the real values and predicted values. The parameters μ_i and σ_i are computed using **Maximum Likelihood Estimation**.

$$\mu_i = \frac{1}{m} \sum_{k=1}^{m} e_{i_j}^k = \bar{e}_{i_j}, \; \sigma_i = \frac{1}{m} \sum_{k=1}^{m} (e_j^k - \mu_j)(e_j^k - \mu_j)^T$$

3.4 Anomaly Detection

In the training stage, the predictive models and estimator modules are periodically used at run-time to infer anomalies. The frequency in which anomaly detection is performed can vary depending on the system specifications. At the run-time, an input measurement x^t is compared with model prediction y^t, and the reconstruction error e^t is calculated. Then, x^t classified as anomalous if $p^t < \alpha$, where p^t is the probability of obtaining the error vector given by the Gaussian estimator G. α is a predefined threshold value, and it is tuned to maximize the F-score of the model.

When anomalous data is discovered, we utilize our consensus algorithm to differentiate between EA and SDA. EA occurs as a result of an incident in the environment. If the EA causes an anomaly in a sensor signal, the correlated sensors are affected by the event and show abnormal changes in their signals. In contrast, SDA influences the sensors individually and results in an anomaly in one or some of the sensors in a cluster. For each cluster, the consensus algorithm inspects the consistency of the sensor behaviors. It uses a voting mechanism to check if all sensors in a cluster agree on the occurrence of an environmental incident. To account for inertia in the physics of the system, we check the consensus in the time intervals instead of data points.

3.5 Model Adaptation

Due to the high variation in the IoT system and environment, we add the property of **aliveness** to our method, which means the model automatically gets updated to adapt to the system alteration and make more accurate predictions. As Figure 3 demonstrates, the sensor association, predictive model, and estimator modules are trainable. There are two levels of updating the model; i)*complete* *update*, which retrains all trainable modules in order, and ii) *partial update*, which only retrains the predictor model. These update processes are triggered under three circumstances:

- Change in the number of sensors in the system (either added or removed) triggers *complete update*.
- Each time the sensors send data, the anomaly detection model first validates the new data. Afterward, *partial update* is triggered using the new anomaly-free data.
- If *complete update* is not provoked during a fixed interval of time $t_{retrain}$, it is triggered automatically. This way, the model accounts for changes in the environment, location, and placement. This parameter $t_{retrain}$ can be tuned by the user, depending on how frequently the system layout is changed.

4 RESULTS AND EVALUATION

4.1 Fog Computing Architecture

Cloud servers are the common and potent available computation resource in IoT systems. However, the bandwidth of network and data transmission become a bottleneck due to Rapid expansion of IoT nodes and the quantity of data. As a result, fog computing has emerged, which provides storage, computation, and application services closer to end-user with dense geographical distribution [29]. In the fog architecture (Figure 7), the bottom layer comprises a heterogeneous network of edge nodes with limited resources. The fog nodes in the middle layer collect and process the data from edge devices and communicate to the cloud via the internet.

Our methodology is fog-empowered, and the developed model for our target IoT system is implemented on a fog node. For the IoT systems with a high density of devices and a massive volume of data, our method is scalable, and it still supports fog computing. Basically, the LSTM encoder-decoder networks are responsible for most of the computation in our method. Thus, instead of training an extensive network for the whole system, we construct a small network for each cluster of associated sensors that can be distributed between fog nodes. Furthermore, we perform several optimizations to meet resource constraints. In the sensor association, we use the binary fingerprint instead of time-series signals, which lowers storage usage and complicity. The sliding window technique in LSTM network contributes to reducing storage usage as well.

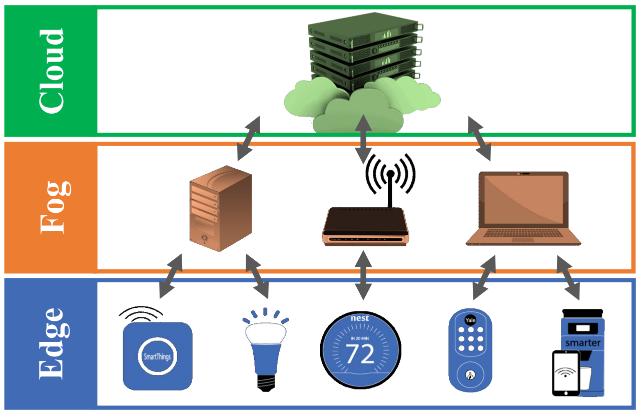
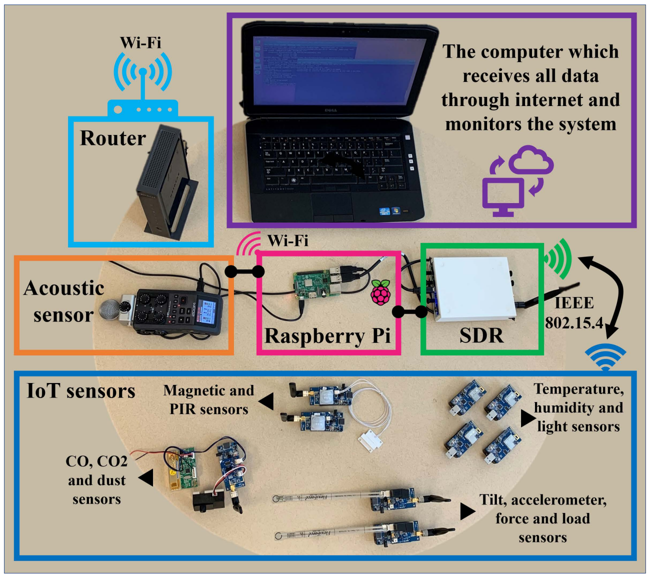


Figure 7: Fog computing hierarchy in IoT systems.


Wi-Fi
Router
The computer which receives all data through internet and monitors the system
Wi-Fi
Acoustic sensor
Raspberry Pi
SDR
IEEE 802.15.4
IoT sensors
Magnetic and PIR sensors
Temperature, humidity and light sensors
CO, CO2 and dust sensors
Tilt, accelerometer, force and load sensors
Figure 8: The scaled-down version of experimental setup.

4.2 Experimental Setup

To build and evaluate our methodology, we implement an IoT testbed in our laboratory. Our experimental setup consists of an Ad-Hoc network of multi-modality IoT sensors, a Software-Defined Radio (SDR) connected to an edge computing device, a gateway, and a laptop as fog node . For this particular research, we have used 62 sensors which measure 13 different physical parameters (see Table 1). The acoustic sensor is a wide range microphone with two right and left channels that captures the sound of the space and its output is amplified and recorded by the handy recorder ZOOM-H6. The raspberry pi board, which is directly connected to ZOOM-H6, collects its data and transmits it over the Internet and this part of the system simulates devices such as Google Home or Alexa. The other sensors are on the low-power embedded boards operated by TinyOS which are equipped with a wireless communication module based on IEEE 802.15.4 standard. We have implemented the IEEE 802.15.4 standard in the SDR device (USRP-B210) and created a

Table 1: List of sensors in our experimental setup.

Sensor	Sensor board	# of sensors
Temperature	MTS-CM5000	12
Humidity	MTS-CM5000	12
Visible light	MTS-CM5000	12
Infrared light	MTS-CM5000	12
Force and load	MTS-CO1000	2
Tilt	MTS-CO1000	2
Accelerometer	MTS-CO1000	2
Presence detector	MTS-SE1000	2
Magnetic	MTS-SE1000	1
CO_2	MTS-AR1000	1
CO	MTS-AR1000	1
Dust	MTS-SH3000	1
Acoustic	ZOOM-H6	2

wireless network of sensors in which SDR collects the sensor's data and send commands to them. SDR is connected to an edge computing device, a raspberry pi board, which works as a base station and gathers all data. The base station contains a Wi-Fi module and links the local network of IoT devices to the Internet through a router. It provides the system with the capability to be monitored in any device which is connected to the internet by looking up the base station and logging using the password. The algorithms and anomaly detection model are implemented on a Laptop with 8Gb DDR4 RAM, and the Intel(R)Core(TM) i5-6300HQ 2.3GHz processor which receives the data from base station and do the computations as a fog node in the IoT system. A powerful router such as Qotom Mini PC Q500G6 has similar capabilities and is capable of running the model at the gateway level. Figure 8 demonstrates the components of our experimental setup and their connections.

4.3 Evaluation

We evaluate our methodology using the data collected by the sensor layout of Section 4.2.

4.3.1 ***Sensor Association Evaluation.*** One of the contributions of our clustering algorithm is the capability to automatically tune the number of clusters and remove the ones which lack a sufficient number of sensors or have sensors that are far apart regarding the hamming distance between their fingerprints. Initially, we set the number of clusters to 20 in our system under test, and the algorithm reduces the number to 6. In order to assess the performance of sensor association method, *Inter-cluster* and *Intra-cluster* distances are calculated for all clusters and plotted in Figure 9. The notable difference between inter-cluster distance and the intra-cluster distance indicates that related sensors are clustered together, and the clusters are well separated from each other.

Another validation method used is physical intuition, which explains the relationships among the associated sensors. For example, co-located sensors experience a similar context. Therefore, they are expected to be associated with each other. This intuition supports the result of our algorithm in which co-located humidity, temperature, and light sensors are clustered together, as it is shown in Figure 11. Another intuition behind the fact is that any physical process may have multi-modality emissions, and the sensors which capture the emission of one incident should be clustered together. It explains the clustering of PIR, vibration sensor (accelerometer and force), magnetic door switch, and acoustic sensor since they all capture the event of entrance through the door. These observations

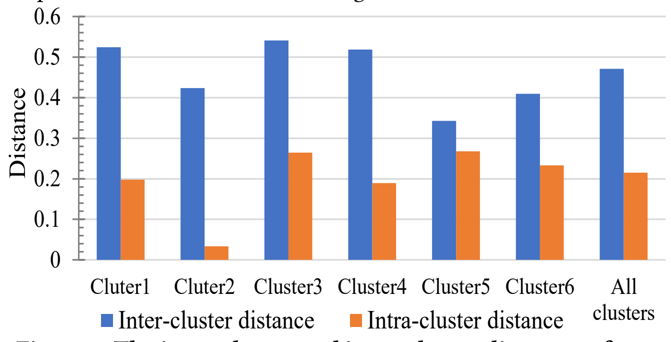


Figure 9: The inter-cluster and intra-cluster distances of sensor clusters.

Table 2: Comparison with the state-of-art methods

Method	Base Model	Context Aware	Precision	Recall	$F_{0.5}$ score	F_1 score
IoT-CAD	LSTM	Yes	%92	%56	%81	%70
[31]	LSTM	No	%64	%44	%58	%52
[28]	Conv LSTM	No	%51	%95	%56	%66
[30]	One Class SVM	No	%89	%25	%60	%39

indicate that this strategy is capable of finding relations between sensors with similar contextual variations, further confirmed by the anomaly detection results in the next section.

4.3.2 ***Anomaly Detection Evaluation.*** The anomaly detection model is unsupervised and it is trained only on the normal data and evaluated using a validation dataset with synthetic anomalies. To analyze the results, True Positives (*TP*), False Positives (*FP*), and False Negatives (*FN*) are counted in the results to compute the validation scores. Although the most intuitive performance measure is accuracy, which is the ratio of correctly predicted observation to the observations, it is not appropriate for unbalanced datasets such as anomaly detection where one category representing the overwhelming majority of the data points. Therefore, we use the *Precision(P), Recall(R)* and F_β *score* as performance metrics.

$$P = \frac{TP}{TP + FP}, R = \frac{TP}{TP + FN}, F_\beta score = \frac{P \times R \times (1 + \beta^2)}{\beta^2 \times P + R}$$

Recall expresses the ability to find all anomalous observation in a dataset while precision expresses the proportion of the observations our model labels as anomaly, actually is anomalous. F_β *score* is the weighted average of precision and recall which provides a better intuition toward both key important capability of model. We implement the current state-of-art methods for anomaly detection in time-series data. Due to importance of precision, $F_{0.5}$ *score* which favors precision over recall is calculated for evaluation in addition to F_1 *score*. According to the results in Table 2, our methodology has the best performance with highest *F scores* and precision.

4.4 Robustness

We evaluate the robustness of our methodology by adding three different types -pink, Gaussian, and uniform- of noise signals to the sensor measurements and observing the performance of the model. As Figure 10 indicates, although the precision of anomaly detection is decreased as the noise power increases in all models, our model is more resilient to noise and maintains the high precision.

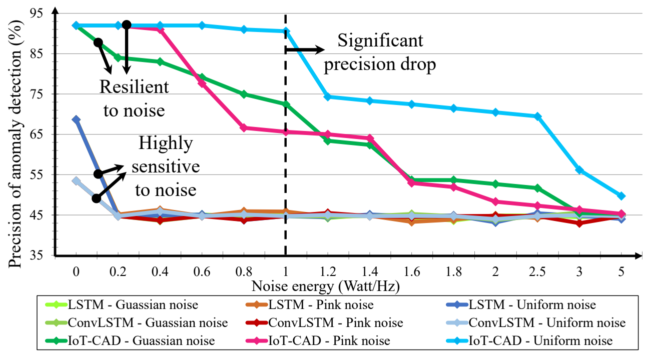


Figure 10: Evaluating the resilience of different models to pink, Gaussian, and uniform noise signals.

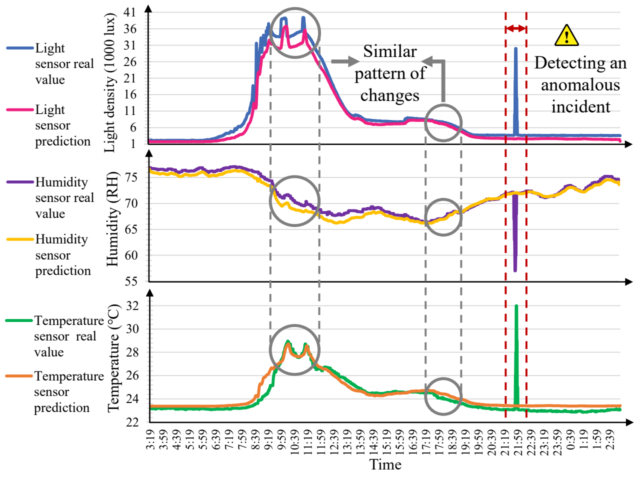


Figure 11: The real and predicted values of three correlated sensors; light, humidity, and temperature sensors.

4.5 Case Study

As a case study, we analyze a cluster of associated sensors, which includes three humidity, temperature, and light sensors located in close proximity. As shown in Figure 11, the predicted values are very close to the real measurements that indicates the competency of our method to learn the normal behavior of sensors and predict the future measurements precisely. Furthermore, we observe that the pattern of changes in the sensor signals is similar. As the marked areas of Figure 11 highlight, any drop in the trend of humidity sensor comes with an increase in the trend of other sensors. It confirms the correlation among the sensors as the sensor association algorithm suggests. We simulate a fire incident in the environment as an EA, and the measurements from all sensors show an anomaly.

4.6 Timing Analysis

The timing of method depends on the number of sensors, length of time-series signals, and computing platform which is used to implement the model. We implement our methodology on a fog computing platform and train it on data collected from 62 heterogeneous sensors for 8 days (roughly, 2.3 million data measurements). The training stage starts with the fingerprint generation process, which is repeated for all sensors (62 times). The sensor association process involves 604 times performing clustering to cluster the patterns and then sensors. Eventually, the clustering layout and sensor measurements are used for training the predictive model in an iterative process until the convergence of the model. Although the initial training is time-consuming, it occurs once, and the process of anomaly detection on the new measurements using the trained model only takes 0.532 seconds, which means it is real-time in our system under test. As mentioned in Section3.5, the retraining process is triggered under some conditions, but it is faster than initial training since it is limited to new data and does not interrupt the anomaly detection (Refer to Table 3).

Table 3: Timing results.

Process	Recurrence	Time (seconds)
Fingerprint generation	62	2.98
Training sensor association	604	10.54
Training predictive model	1	2472.20
Anomaly detection	periodic	0.532

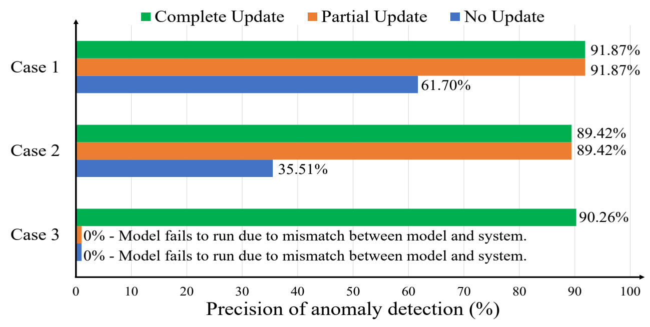


Figure 12: Analysis of effect of partial and complete update on preserving the performance of model.

4.7 Aliveness Assessment

To assess whether updating the model is beneficial for maintaining the model's performance over time, we test it in three scenarios. The first and second scenarios simulate the effect of system degradation or environmental variation over time. In this regard, the measurements of temperature sensors are increased 5^{o}C in case 1, and 15^{o}C in case 2 for a day. The third scenario simulates changes in the layout of the IoT system by eliminating a sensor.

The model is initially trained on the original data before the occurrence of scenarios and tested with synthesized data from the cases. In the tests, we examine the effect of the partial update, complete update, and no update on the precision of the model, refer to Figure 12. According to results, the variation in case 1 and 2 lead to a significant drop in the precision of the model without updating while updating the model, effectively preserve the high performance because of retraining the predictive model. The third case highlights the advantage of the complete update. Any alteration (add or remove) in the number of sensors in the IoT system changes the input layer dimension of the neural network. Thus, the model cannot perform anomaly detection in case 3 unless the sensor association is retrained to update the layout of sensors, and the model is reconstructed on the new layout. Results confirm that the complete update is successful in maintaining the performance despite removing a sensor. Based on this experiment, it can be concluded that being adaptive is crucial for the models used for IoT.

5 CONCLUSION

This paper presents a novel context-aware adaptive data-driven model for anomaly detection in IoT systems. It generates context information by encoding the relations among the IoT sensors and clusters the correlated sensors based upon similar patterns, and contextual variation. According to the extracted context, a predictive model detects the anomalies, and a consensus-based algorithm determines the type of detected anomalies and pinpoints their source. Our proposed methodology can identify the anomalies with a **92%** precision in real-time on a fog computing platform. Compared with other methods, it has higher performance and the capability to update itself to account for variations in the system and environment.

ACKNOWLEDGMENTS

This research was supported by NSF award CMMI 1739503 and ONR award N00014-17-1-2499. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect views of our funding agencies.

REFERENCES

[1] M. M. Ahemd, M. A. Shah, and A. Wahid. 2017. IoT security: A layered approach for attacks amp; defenses. In *2017 International Conference on Communication Technologies (ComTech)*. 104–110. https://doi.org/10.1109/COMTECH.2017.8065757

[2] Unai Alegre, Juan Carlos Augusto, and Tony Clark. 2016. Engineering context-aware systems and applications: A survey. *Journal of Systems and Software* 117 (2016), 55–83.

[3] D. Altolini, V. Lakkundi, N. Bui, C. Tapparello, and M. Rossi. 2013. Low power link layer security for IoT: Implementation and performance analysis. In *2013 9th International Wireless Communications and Mobile Computing Conference (IWCMC)*. 919–925. https://doi.org/10.1109/IWCMC.2013.6583680

[4] Anomadarshi Barua and Mohammad Al Faruque. 2020. Hall Spoofing: A Non-Invasive DoS Attack on Grid-Tied Solar Inverter. *29th Usenix Security* (2020).

[5] Ferdinand Brasser, Brahim El Mahjoub, Ahmad-Reza Sadeghi, Christian Wachsmann, and Patrick Koeberl. 2015. TyTAN: tiny trust anchor for tiny devices. In *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. IEEE, 1–6.

[6] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. 2000. LOF: identifying density-based local outliers. In *ACM sigmod record*, Vol. 29. ACM, 93–104.

[7] Sucheta Chauhan and Lovekesh Vig. 2015. Anomaly detection in ECG time signals via deep long short-term memory networks. In *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*. IEEE, 1–7.

[8] Hsin Chung Chen, Mohammad Abdullah Al Faruque, and Pai H Chou. 2016. Security and privacy challenges in IoT-based machine-to-machine collaborative scenarios. In *Proceedings of the Eleventh IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*.

[9] Sujit Rokka Chhetri, Sina Faezi, Arquimedes Canedo, and Mohammad Abdullah Al Faruque. 2019. QUILT: quality inference from living digital twins in IoT-enabled manufacturing systems. In *Proceedings of the International Conference on Internet of Things Design and Implementation*. ACM, 237–248.

[10] Sujit Rokka Chhetri, Nafiul Rashid, Sina Faezi, and Mohammad Abdullah Al Faruque. 2017. Security trends and advances in manufacturing systems in the era of industry 4.0. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*.

[11] Andrew Cook, Göksel Mısırlı, and Zhong Fan. 2019. Anomaly Detection for IoT Time-Series Data: A Survey. *Internet of Things Journal* (2019).

[12] Sina Faezi, Sujit Rokka Chhetri, Arnav Vaibhav Malawade, John Charles Chaput, William H Grover, Philip Brisk, and Mohammad Abdullah Al Faruque. 2019. Oligo-Snoop: A Non-Invasive Side Channel Attack Against DNA Synthesis Machines.. In *NDSS*.

[13] Pavel Filonov, Andrey Lavrentyev, and Artem Vorontsov. 2016. Multivariate Industrial Time Series with Cyber-Attack Simulation: Fault Detection Using an LSTM-based Predictive Data Model. arXiv:1612.06676 [cs.LG]

[14] Federico Giannoni, Marco Mancini, and Federico Marinelli. 2018. Anomaly Detection Models for IoT Time Series Data. *arXiv* (2018).

[15] B. Halak, M. Zwolinski, and M. S. Mispan. 2016. Overview of PUF-based hardware security solutions for the internet of things. In *2016 IEEE 59th International Midwest Symposium on Circuits and Systems (MWSCAS)*. 1–4. https://doi.org/10.1109/MWSCAS.2016.7870046

[16] Jun Han, Albert Jin Chung, Manal Kumar Sinha, Madhumitha Harishankar, Shijia Pan, Hae Young Noh, Pei Zhang, and Patrick Tague. 2018. Do you feel what I hear? Enabling autonomous IoT device pairing using different sensor types. In *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 836–852.

[17] Mee Lan Han, Jin Lee, Ah Reum Kang, Sungwook Kang, Jung Kyu Park, and Huy Kang Kim. 2015. A Statistical-Based Anomaly Detection Method for Connected Cars in Internet of Things Environment. In *Internet of Vehicles - Safe and Intelligent Mobility*, Ching-Hsien Hsu, Feng Xia, Xingang Liu, and Shangguang Wang (Eds.). Springer International Publishing, Cham, 89–97.

[18] Zengyou He, Xiaofei Xu, and Shengchun Deng. 2003. Discovering cluster-based local outliers. *Pattern Recognition Letters* 24, 9-10 (2003), 1641–1650.

[19] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[20] E. Hodo, X. Bellekens, A. Hamilton, P. Dubouilh, E. Iorkyase, C. Tachtatzis, and R. Atkinson. 2016. Threat analysis of IoT networks using artificial neural network intrusion detection system. In *2016 International Symposium on Networks, Computers and Communications (ISNCC)*. 1–6. https://doi.org/10.1109/ISNCC.2016.7746067

[21] C. Lesjak, H. Bock, D. Hein, and M. Maritsch. 2016. Hardware-secured and transparent multi-stakeholder data exchange for industrial IoT. In *2016 IEEE 14th International Conference on Industrial Informatics (INDIN)*. 706–713. https://doi.org/10.1109/INDIN.2016.7819251

[22] Christian Lesjak, Norbert Druml, Rainer Matischek, Thomas Ruprechter, and Gerald Holweg. 2016. Security in industrial IoT – quo vadis? *e & i Elektrotechnik und Informationstechnik* 133, 7 (01 Nov 2016), 324–329. https://doi.org/10.1007/s00502-016-0428-4

[23] Tao Lin, Tian Guo, and Karl Aberer. 2017. Hybrid Neural Networks for Learning the Trend in Time Series. (2017), 2273–2279. https://doi.org/10.24963/ijcai.2017/316

[24] Wei-Chao Lin, Shih-Wen Ke, and Chih-Fong Tsai. 2015. CANN: An intrusion detection system based on combining cluster centers and nearest neighbors. *Knowledge-based systems* 78 (2015), 13–21.

[25] Y. Liu, H. Zheng, X. Feng, and Z. Chen. 2017. Short-term traffic flow prediction with Conv-LSTM. In *2017 9th International Conference on Wireless Communications and Signal Processing (WCSP)*. 1–6. https://doi.org/10.1109/WCSP.2017.8171119

[26] Barthélémy Longueville, Mickaël Gardoni, et al. 2003. A survey of context modeling: approaches, theories and use for engineering design researches. In *DS 31: Proceedings of ICED 03, the 14th International Conference on Engineering Design, Stockholm*. 437–438.

[27] YANG Zhihong TU Mengfu LU Jinjun LU Jixiang, ZHANG Qipei and PENG Hui. 2019. Short-term Load Forecasting Method Based on CNN-LSTM Hybrid Neural Network Model. *AEPS* 43, 8 (2019), 131. https://doi.org/10.7500/AEPS20181012004

[28] W. Luo, W. Liu, and S. Gao. 2017. Remembering history with convolutional LSTM for anomaly detection. In *2017 IEEE International Conference on Multimedia and Expo (ICME)*. 439–444. https://doi.org/10.1109/ICME.2017.8019325

[29] Lingjuan Lyu, Jiong Jin, Sutharshan Rajasegarar, Xuanli He, and Marimuthu Palaniswami. 2017. Fog-empowered anomaly detection in IoT using hyperellipsoidal clustering. *Internet of Things Journal* (2017).

[30] Junshui Ma and Simon Perkins. 2003. Time-series novelty detection using one-class support vector machines. In *International Joint Conference on Neural Networks*.

[31] Pankaj Malhotra, Anusha Ramakrishnan, Gaurangi Anand, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. 2016. LSTM-based encoder-decoder for multi-sensor anomaly detection. *arXiv preprint arXiv:1607.00148* (2016).

[32] Pankaj Malhotra, Vishnu TV, Anusha Ramakrishnan, Gaurangi Anand, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. 2016. Multi-Sensor Prognostics using an Unsupervised Health Index based on LSTM Encoder-Decoder. arXiv:1608.06154 [cs.LG]

[33] Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal. 2015. Long short term memory networks for anomaly detection in time series. In *Proceedings*. Presses universitaires de Louvain, 89.

[34] Jonathan M McCune, Yanlin Li, Ning Qu, Zongwei Zhou, Anupam Datta, Virgil Gligor, and Adrian Perrig. 2010. TrustVisor: Efficient TCB reduction and attestation. In *2010 IEEE Symposium on Security and Privacy*. IEEE, 143–158.

[35] Markus Miettinen, N Asokan, Thien Duc Nguyen, Ahmad-Reza Sadeghi, and Majid Sobhani. 2014. Context-based zero-interaction pairing and key evolution for advanced personal devices. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 880–891.

[36] Markus Miettinen, Thien Duc Nguyen, Ahmad-Reza Sadeghi, and N Asokan. 2018. Revisiting context-based authentication in IoT. In *2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC)*. IEEE, 1–6.

[37] Yao Qin, Dongjin Song, Haifeng Chen, Wei Cheng, Guofei Jiang, and Garrison Cottrell. 2017. A Dual-Stage Attention-Based Recurrent Neural Network for Time Series Prediction. arXiv:1704.02971 [cs.LG]

[38] Shahid Raza, Linus Wallgren, and Thiemo Voigt. 2013. SVELTE: Real-time intrusion detection in the Internet of Things. *Ad hoc networks* 11, 8 (2013), 2661–2674.

[39] H. Sedjelmaci, S. M. Senouci, and M. Al-Bahri. 2016. A lightweight anomaly detection technique for low-resource IoT devices: A game-theoretic methodology. In *2016 IEEE International Conference on Communications (ICC)*. 1–6. https://doi.org/10.1109/ICC.2016.7510811

[40] Arvind Seshadri, Mark Luk, and Adrian Perrig. 2008. SAKE: Software Attestation for Key Establishment in Sensor Networks. In *Distributed Computing in Sensor Systems*, Sotiris E. Nikoletseas, Bogdan S. Chlebus, David B. Johnson, and Bhaskar Krishnamachari (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 372–385.

[41] Arvind Seshadri, Mark Luk, Adrian Perrig, Leendert van Doorn, and Pradeep K. Khosla. 2006. SCUBA: Secure Code Update By Attestation in sensor networks. In *Workshop on Wireless Security*.

[42] Jiang Wan, Anthony Lopez, and Mohammad Abdullah Al Faruque. 2018. Physical layer key generation: Securing wireless communication in automotive cyber-physical systems. *ACM Transactions on Cyber-Physical Systems* 3, 2 (2018), 13.

[43] Yuankai Wu and Huachun Tan. 2016. Short-term traffic flow forecasting with spatial-temporal correlation in a hybrid deep learning framework. arXiv:1612.01022 [cs.CV]

[44] Haiyang Yu, Zhihai Wu, Shuqin Wang, Yunpeng Wang, and Xiaolei Ma. 2017. Spatiotemporal Recurrent Convolutional Networks for Traffic Prediction in Transportation Networks. *Sensors* 17, 7 (2017). https://doi.org/10.3390/s17071501

[45] K. Zhao and L. Ge. 2013. A Survey on the Internet of Things Security. In *2013 Ninth International Conference on Computational Intelligence and Security*. 663–667. https://doi.org/10.1109/CIS.2013.145

[46] Z. Zhao, W. Chen, X. Wu, P. C. Y. Chen, and J. Liu. 2017. LSTM network: a deep learning approach for short-term traffic forecast. *IET Intelligent Transport Systems* 11, 2 (2017), 68–75. https://doi.org/10.1049/iet-its.2016.0208