Open Access

Open Journal of Internet of Things (OJIOT)
Volume 8, Issue 1, 2022

www.ronpub.com/ojiot
ISSN 2364-7108

3D Histogram Based Anomaly Detection for Categorical Sensor Data in Internet of Things

Peng Yuan[A], Lu-An Tang[A], Haifeng Chen[A], Moto Sato[A], Kevin Woodward[B]

[A] NEC Labs America, 4 Independence Way, Princeton, NJ, USA, {pyuan, ltang, Haifeng, moto}@nec-labs.com
[B] Lockheed Martin Space, 1401 Del Norte St, Denver, CO 80221, USA, {kevin.woodward}@lmco.com

ABSTRACT

The applications of Internet-of-things (IoT) deploy massive number of sensors to monitor the system and environment. Anomaly detection on streaming sensor data is an important task for IoT maintenance and operation. In real IoT applications, many sensors report categorical values rather than numerical readings. Unfortunately, most existing anomaly detection methods are designed only for numerical sensor data. They cannot be used to monitor the categorical sensor data. In this study, we design and develop a 3D Histogram based Categorical Anomaly Detection (HCAD) solution to monitor categorical sensor data in IoT. HCAD constructs the histogram model by three dimensions: categorical value, event duration, and frequency. The histogram models are used to profile normal working states of IoT devices. HCAD automatically determines the range of normal data and anomaly threshold. It only requires very limit parameter setting and can be applied to a wide variety of different IoT devices. We implement HCAD and integrate it into an online monitoring system. We test the proposed solution on real IoT datasets such as telemetry data from satellite sensors, air quality data from chemical sensors, and transportation data from traffic sensors. The results of extensive experiments show that HCAD achieves higher detecting accuracy and efficiency than state-of-the-art methods.

TYPE OF PAPER AND KEYWORDS

Application paper: *Internet of Things, sensor data, anomaly detection, data stream*

1 INTRODUCTION

Internet of things (IoT) integrates sensor devices with informational components to form a context sensitive system that responds intelligently to dynamic changes in real-world environments. With rapid development in recent years, IoT devices are widely used in different fields such as social media, healthcare, transportation, satellite, and environment monitoring. A typical IoT application usually contains massive number of sensors to monitor related systems or surrounding environment. Evaluating streaming data in real-time and detecting abnormal symptoms are critical for system maintenance and operation. Efficient and effective anomaly detection methods are required to increase engineering productivity and reduce maintenance cost for IoT.

However, the problems of online data monitoring and anomaly detection are one of the most difficult

This paper is accepted at the International Workshop on Very Large Internet of Things (VLIoT 2022) in conjunction with the 2022 VLDB Conference in Sydney, Australia. The proceedings of VLIoT@VLDB 2022 are published in the Open Journal of Internet of Things (OJIOT) as special issue.

problems in IoT research fields [10], partly due to the following challenges:

- Categorical value: In real applications, IoT sensors contain not only numerical readings but also categorical or symbolic data representing the working status and operational modes. In recent decades, researchers proposed various technologies to detect anomalies on numerical readings [3-5]. Unfortunately, anomaly detection in categorical data has received little attention from the research community. It is also not easy to extend the anomaly detection methods on numerical values to categorical data. Because such methods focus on detecting extreme values or outliers. However, most categorical anomalies are not related to the values and cannot be detected by existing methods.
- Real time detection: In several IoT applications, the actions must be taken out immediately to deal with anomalies. Many methods are in the style of offline learning [4], they cannot detect anomalies in real time.

In this study, we propose a novel solution called 3D Histogram based Categorical Anomaly Detection (HCAD) for IoT systems. In this solution, a 3D histogram model is obtained from the historical data of categorical sensors. The model's three dimensions are: categorical value, event duration, and frequency. HCAD profiles the normal state by such model, and automatically determines the normal value range despite of noisy data. After training the model from historical data, HCAD monitors the newly arrived data and matches them with historical model. The anomaly score is computed based on the model and anomaly detection is made in real time. The contributions of this study are summarized as follows.

- We propose and define the problem of anomaly detection on categorical sensor data, which is rarely studied but important for real IoT applications.
- We design and develop a solution of 3D Histogram based Categorical Anomaly Detection (HCAD). HCAD has low computation cost and can be easily aggregated into various IoT monitoring systems.
- We conduct extensive experiments to evaluate the effectiveness and efficiency of HCAD on public datasets from real world. The experiment results show that HCAD yields better performance than state-of-the-art methods.
- We integrate HCAD into a IoT monitoring system. The monitoring system only needs very limited parameter setting and can be applied to monitor a wide variety of IoT devices.

The remaining sections of this paper are organized as follows. Section 2 surveys the related work of anomaly detection in IoT. Section 3 defines the problem and introduces the system framework. Section 4 explains the proposed solution in detail. Section 5 presents the experiments of testing proposed solution and other baselines. Finally, Section 6 summarizes the paper and discusses future work.

2 RELATED WORK

In recent decades, the problem of anomaly detection on data stream and sensor readings have received widespread attentions from research communities. Many methods are proposed to detect outliers and anomalies on sensor data. These techniques can be roughly classified into two classes: knowledge-driven approaches and data-driven methods.

2.1 Knowledge-Driven Approaches

Knowledge-driven approaches are based on the evidence of user guidelines and expert's knowledge.

Shabtai [1] proposed anomaly detection method based on Knowledge-Based Temporal Abstraction (KBTA). This method was previously proposed for intelligent interpretation of temporal data based on predefined domain knowledge [2]. The method uses a temporal pattern mining process to extract patterns representing normal behavior. Some researchers try to monitor the sensor data by predefined models. In practical applications, the sensor data may have some regular patterns. If such patterns can be predefined by expert's knowledge, it is much easier to detect anomalies based on the models. Tang et al. define a Pseudo Periodical Graph (PPG) model to detect anomalies on medical sensors [7].

Knowledge-driven methods can explain anomalies in detail if the knowledge from experts is accurate and comprehensive. However, with the rapid development of IoT, it is difficult to obtain accurate knowledge to describe complex and large scale IoT systems.

2.2 Data-Driven Methods

In the past few years, many data-driven methods are proposed for anomaly detection, including regression based model [3], clustering based model [4][15] and so on.

Table 1: Summary of anomaly detection methods

Method	Prior knowledge	Categorical data	Online monitor
KBTA [1]	Yes	No	No
PPG [7]	Yes	No	Yes
Regression tree [3]	No	No	No
Clustering model [4]	No	No	No
LSTM-NDT [5]	No	No	Yes
HCAD	No	Yes	Yes
CUNA [15]	No	No	No
RobustTAD [16]	No	No	No
CNNTL [17]	No	No	No

Yairi et.al propose an anomaly detection method to monitor health status of spacecrafts based on regression tree learning [3]. Gao et.al design an unsupervised anomaly detection approach for spacecraft based on normal behavior clustering [4]. The method takes unlabeled historical telemetry data as input and generates a behavior model by extracting clusters from the normal data. Aytekin et.al propose a clustering based unsupervised anomaly detection method using L-2 normalized deep auto-encoder representations [15]. The method maps past normal data to the high dimensional feature space using kernel function. In testing stage, the method projects new data to the learned subspace and calculates anomaly degree by distribution model. More recently, Neural Networks based techniques have demonstrated good performance on anomaly detection on streaming data. Hundman et.al propose an unsupervised anomaly detection approach based on long short term memory networks (LSTM-NDT) [5]. Gao et.al propose RobustTAD, a robust time series anomaly detection framework by integrating robust time series decomposition and convolutional neural network for time series data [16]. Wen et.al use a time series segmentation approach based on convolutional neural networks for anomaly detection. Moreover, they also propose a transfer learning framework. The framework pre-trains a model on large-scale synthetic univariate time series data and then fine-tunes the weights on small-scale, univariate, or multivariate data sets with previously unseen classes of anomalies [17].

The major advantage of data-driven methods is that they do not need domain knowledge in advance. In addition, they can easily be applied to different scenarios. However, most existing approaches are designed only for numerical sensor data. They cannot be used for categorical data. Our Histogram based Categorical Anomaly Detection (HCAD) solution is thus proposed to monitor the health of IoT devices with categorical sensor readings.

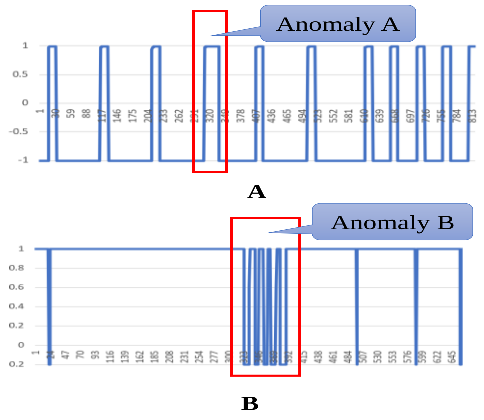


Figure 1: Categorical sensor data from satellite

Table 1 presents a summary of the related methods in different categories, comparing to the proposed HCAD in multiple criteria.

3 OVERVIEW

This section presents the problem definition of anomaly detection on categorical data, as well as the system framework of HCAD.

3.1 Task specification

Satellite health monitoring is a typical application case of categorical anomaly detection in IoT. We motivate the problem in accordance with the requirements of real-world satellite monitoring tasks.

Figure 1 shows recorded data from two sensors. Sensor s_a and Sensor s_b are two sensors installed on a telemetry satellite. They are used to monitor the satellite's working status. There are four key observations:

1. **Categorical data:** There is a set of unique values and the sensor reading must be one of these values. The value set of sensor s_a is V_a = {-1, -0.5, 0, 0.5, 1}, the value set of sensor s_b is V_b = {-0.2, 0, 0.2, ..., 1}.
2. **No anomaly of extreme value:** All the values appear in the sensor's data and none of them is related to any anomaly. There are multiple peaks and valleys in Figure 1, but most of them are just normal cases.
3. **Anomaly of duration:** The red block of Figure 1 (A) indicates an abnormal event of the satellite: Sensor A has a working state of value "1" in every 60 secs, and the working state only lasts for about 10 secs. In the red block period, a component is over-heated, and the sensor's working time almost

doubled. The event is abnormal because of the duration.

4. **Anomaly of frequency:** The red block of Figure 1 (B) indicates another abnormal event of the satellite. Sensor B is used to monitor a key component of the satellite. Value "1" means the component is activated and value "-0.2" means the component is deactivated. In the red block period, the satellite tried to activate the component but failed multiple times due to a hardware problem. After several attempts in short period, the component is finally activated. This event is indeed a precursor of more serious mechanic problem. It can only be detected based on the event frequency.

Based on the above observations, we formally define the task of anomaly detection in categorical IoT.

Definition 1 (Categorical sensor reading). Let *S* be the set of categorical sensors in a IoT deployment, $S = \{s_1, s_2, \ldots, s_n\}$, and s_i be a categorical sensor. The reading of s_i is denoted as R_i. R_i is a sequence of length *m*, $R_i = \{(v_1, t_1), (v_2, t_2) \ldots (v_m, t_m)\}$, where t_j is the timestamp and v_j is the reading value. $v_j \in V_i$, where V_i is the set of unique values of sensor s_i.

Problem Definition. Let *S* be the set of categorical sensors in a IoT deployment, $S = \{s_1, s_2, \ldots, s_n\}$. Given the historical sensor readings $R_1, R_2, \ldots, R_n$, the system is required to train a model from the historical data and use the trained model to monitor newly arriving data of $s_1, s_2, \ldots, s_n$ and detect anomalies on them.

3.2 System Framework

In this paper, we propose a 3D Histogram based Categorical Anomaly Detection (HCAD) approach to monitor the categorical sensors of IoT and detect anomalies on the streaming data.

As shown in Figure 2, the overall structure of HCAD consists of two parts: (1) offline training from historical data and (2) online monitoring for the streaming data.

Adaptive window separation is a module shared by both offline training and online monitoring. By this module, the categorical sensor data are segmented into sequences through sliding windows of adaptive length. After window separation, HCAD learns the features of all historical segments and generates a 3D histogram model to profile the normal data. At last, HCAD computes the threshold by testing the model on historical normal data.

The online monitoring module stores the trained histogram model. HCAD converts streaming data into a new sliding window, and then transforms the data within sliding window into a histogram. The new histogram is matched with historical model to calculate an anomaly score. The system raises alert if the anomaly score is higher than trained threshold.

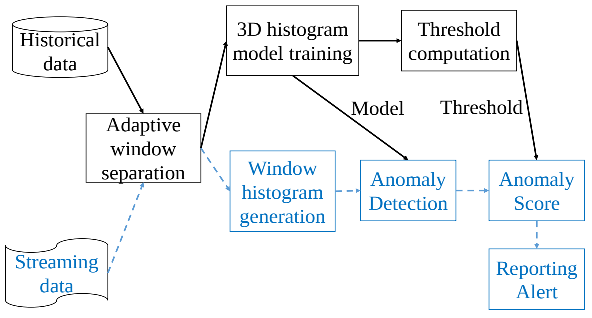


Figure 2: System framework of HCAD. The black solid lines denote the procedure of offline training and the blue dash lines denote steps of online monitoring.

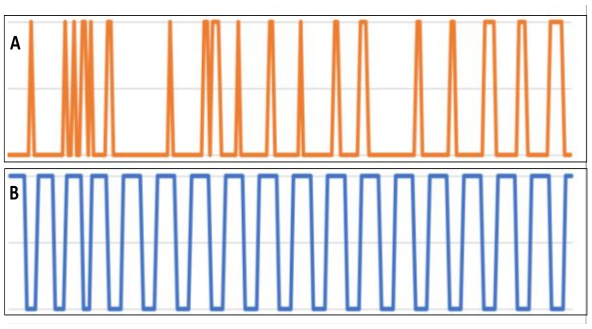


Figure 3: Two categorical sensor data with periodicity

4 Design of HCAD Method

In this section, we present the design of 3D histogram-based anomaly detection method. We first introduce the offline training modules, including adaptive sliding window separation and histogram model training, then present the details of online monitoring and anomaly detection modules.

4.1 Adaptive Sliding Window Separation

Since the historical sensor data are collected in a relatively long time period, i.e., several weeks or months. The system needs to first separate the long data sequence into several sub-sequences, i.e., sliding windows. Here the window length *l* is an important parameter: if *l* is too short, the system may not capture the long events and cause false negatives; if *l* is too long, the system has high memory and computation overhead, and cannot output the alerts in time. To overcome these challenges, we design an adaptive

Algorithm 1: Adaptive sliding window separation

Input: sensor data R, minimum number of event m, sliding speed p (p is set as 1 in most cases)

Output: sliding window sequence W, length of sliding window l

```
1.  initialize window size $l = 0$
2.  event count $c = 0$
3.  total event duration $d_{total} = 0$
4.  **for** each event $e$ in $R$
5.      $d_{total} += (e_{end} - e_{start})$
6.      $c += 1$
7.      **if** $c == m$
8.          **if** $d_{total} > l$ **then** $l = d_{total}$
9.          $c = 0$
10. initialize $W = \{w_1, w_2, \dots w_{(|R|-l)/p}\}$
11. **for** each window $w_i$ in $W$
12.     $t_{start} = p*i$, $t_{end} = p*i + l$
13.     retrieve $R_i = \{(v_{start}, t_{start}), \dots (v_{end}, t_{end})\}$
14.     generate a event sequence $E_i$ from $R_i$
15.     add $E_i$ to $w_i$
16. **return** $W, l$
```

algorithm to determine the window length automatically.

Example 1: Figure 3 shows two categorical readings from satellite sensors. We can make a key observation from the figure: The duration of certain categorical values repeats in a regular way, i.e., there are some kinds of periodicity on the duration of the categorical values.

Definition 2 (Event). Let s be a categorical sensor and R be the sensor readings from s, $R = \{(v_1, t_1), (v_2, t_2) \dots (v_n, t_n)\}$, where t_j is the timestamp and v_j is the reading at t_j. For consecutive readings $v_j, v_{j+1}, \dots, v_{j+k}$, if $v_j = v_{j+1} = \dots = v_{j+k}$, we can merge these readings to event $e_j = (v_j, t_j, t_{j+k})$, where t_j is the event start timestamp and t_{j+k} is the event end timestamp. In this way, R_i is transformed as a sequence of categorical events. $R_i = \{e_1, e_2 \dots e_m\}$, $e_i = (v_i, t_i, t_{i+k})$.

In categorial sensor data, the event is a more important object than a single point. Usually, an event reflects a period of certain work state of the monitored component. The start and end of an event are often related to the system operations (e.g., turn on or turn off a component).

To make a meaningful separation of the sensor data, each sliding window should contain enough number of events. Given the sensor data R_i, minimum number of event m and sliding speed p, Algorithm 1 computes the window length l and generates the sliding window over sensor data. After initializing the parameters (Line 1 - 3), the algorithm scans the sensor data R and computes the total duration for consecutively m events (Lines 4 - 6). The maximum duration is recorded as the window length (Lines 7 - 9). After that, the algorithm initializes the sliding window sequence and scans the data again. The system retrieves subsequences of sensor data by length l and generates the event sequence from the data (Lines 10 to 15). Finally, the algorithm returns the generated window sequence (Line 16).

4.2 Histogram Model Construction

In this step, HCAD needs to generate a histogram model to profile the normal states of the monitored IoT system. Since the unique categorical values indicate different working states of the IoT, HCAD needs to generate a specific histogram for each categorical value and distributes the events into m bins by the event duration. The height of each bin is the count of event (i.e., frequency). As a result, there are 3 dimensions in the histogram: (1) the category, (2) the event duration, and (3) the frequency.

Because the distributions of events' duration are varied on different cases. It is necessary to generate an adaptive histogram. In most cases, the data of events' duration are not in a standard gaussian distribution. Hence the difficulty is on identifying the optimal mean and standard deviation for events' duration. The Weibull distribution is a continuous probability distribution widely used to analyze event data, model failure times and measure product reliability [13]. The major advantage of Weibull distribution over other ones is on the flexibility. Weibull distribution can be used to simulate other distributions. Therefore, we use Weibull distribution to approximate the distribution of events' duration and subsequently compute optimal mean and standard deviation. As shown in Formula 1, HCAD determines the value range of histogram model based on optimal mean and standard deviation.

$$
\begin{aligned}
Min\_Bound &= \begin{cases} \mu - \alpha * \sigma, & if\ \mu - \alpha_1 * \sigma > 0 \\ 0, & if\ \mu - \alpha_1 * \sigma \le 0 \end{cases} \\
Max\_Bound &= \begin{cases} \mu + \beta * \sigma, & if\ \mu + \beta_1 * \sigma < L \\ L, & if\ \mu + \beta_1 * \sigma \ge L \end{cases}
\end{aligned}
\tag{1}
$$

Here, Min_Bound represents the lower bound of the value range in the histogram. Max_Bound represents the upper bound of the value range in the histogram. L is length of sliding window. μ is mean of all the events' duration. σ is the duration's standard deviation. β_1 and α_1 are two parameters.

Once the value range of histogram is determined, HCAD further divides the value range into m equally distant bins and generates the histogram template. Algorithm 2 describes the process of generating histogram template for each category. The algorithm first Initializes the parameters (Lines 1 to 5), then

Algorithm 2: 3D histogram template generation

Input: Unique category value list U, number of bins m, parameters α_1, β_1, sliding window sequence W, length of sliding window l

Output: 3D histogram template T

```
1.  $T = \emptyset$
2.  **for** each unique category value $v_u$ in $U$
3.      Min_Bound(u) $= 0$
4.      Max_Bound (u) $= l$
5.      $D_u = \emptyset$
6.      **for** each window $w$ in $W$
7.          **for** each event $e$ in $w$
8.              **if** $v_e = v_u$
9.                  $D_u$.append($e$)
10.     use Weibull distribution to model $D_u$
11.     compute $\mu$ and $\sigma$ based on $D_u$'s distrubution
12.     **if** $\mu - \alpha * \sigma > 0$
13.         Min_Bound(u) $= \mu - \alpha_1 * \sigma$
14.     **if** $\mu + \beta * \sigma < l$
15.         Max_Bound (u) $= \mu + \beta_1 * \sigma$
16.     generate 2D histogram template $h_u$
17.     $T$.append($h_u$)
18. **return** $T$
```

generates the event list for each category value from sliding window sequence (Lines 6 to 9). After that, the algorithm uses Weibull distribution to approximate the data of each event list and determines the value range of histogram template based on optimal mean and standard deviation (Lines 10 to 15). Finally, the system generates a histogram template for the specific category, adds the template to histogram set and return the set after processing all the categories (Lines 16 to 18). After the 3D histogram template is generated, HCAD fills the template with the events in sliding window sequence.

Example 2: Figure 4 shows an example for filling the template with events to generate histograms. There are two categories (category A and B) in the data. HCAD first generates two histogram templates. Two sliding windows are shown in Figure 4 (w_1 in red color and w_2 in yellow color). Two 3D histogram, h_1 and h_2 are generated for w_1 and w_2. In w_1, there are 9 events retrieved from the sensor data. e_1, e_3, e_5, e_7 and e_9's values are same to category A. They are matched to the histogram h_1^A. While e_2, e_4, e_6, and e_8's values are same to category B. They are matched to histogram h_1^B. Similarly, e_5, e_7, e_9, e_{11} and e_{13} of window w_2 are matched to histogram h_2^A. e_6, e_8, e_{10} and e_{12} of w_2 are matched to histogram h_2^B.

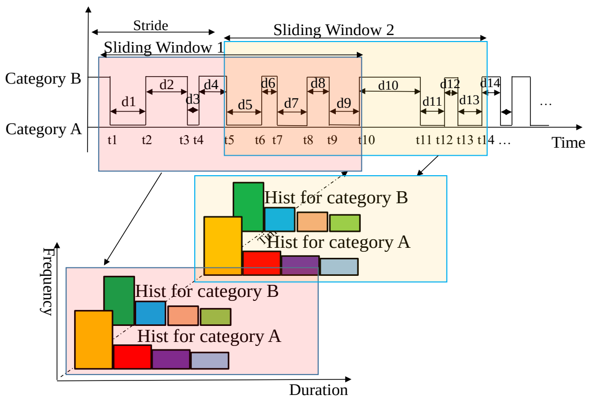


Figure 4: Generating histogram from sliding window

In this way, HCAD generates a series of 3D histograms from the historical data. By concatenating all these 3D histograms, we can get a $k * n * m$ tensor, where k is the size of unique category, n is the total number of sliding windows on historical data and m is the number of bins in histogram template. The value stored in the tensor is the count of event's duration (i.e., frequency) in each bin.

The next step is to learn a model for every bin in each category. To learn the model for category v_i and bin b_j, HCAD makes a slice from the tensor to get a time series of n values by v_i and b_j. This time series records all the frequencies in the historical sliding window. HCAD uses a Weibull distribution to approximate the time series, and computes the optimal mean and standard deviation. At last, HCAD determines the normal ranges of bin b_j in category v_i's model by these parameters, as shown in Eq. (2), where $lb(v_i, b_j)$ and $ub(v_i, b_j)$ are the lower bound and upper bound of the normal range for frequency. $\mu_{i,j}$ is mean and $\sigma_{i,j}$ is the standard deviation computed from the Weibull distribution. α_2 and β_2 are two parameters.

$$\begin{aligned} lb(v_i, b_j) &= \mu_{i,j} - \alpha_2 * \sigma_{i,j} \\ ub(v_i, b_j) &= \mu_{i,j} + \beta_2 * \sigma_{i,j} \end{aligned} \tag{2}$$

Algorithm 3 shows the process to compute the histogram model. The algorithm takes the 3D histogram template as input. For each categorical value, the algorithm retrieves the 2D histogram template and fills it with the events in sliding windows of historical data (Lines 1 to 4). Then the algorithm retrieves the frequency vector by bin and category (Lines 5 to 6), and uses the Weibull distribution to compute the mean and standard deviation (Lines 7 to 8). Furthermore, the algorithm computes the range for frequency and fills such information to the model (Lines 9 to 11). At last, the system returns the trained model (Line 12).

Algorithm 3: Histogram model computation

Input: 3D histogram template T, parameters α_2, β_2, sliding window sequence W, number of bins m

Output: Histogram model H

```
1.  for each categorical value $v_u$ in $T$
2.      2D histogram $h_u$ = $T.get(v_u)$
3.      for each window $w$ in $W$
4.          fill $h_u$ with the events in $w$
5.      for each bin $b_i$ in $h_u$
6.          retrieve the frequency vector $V_{i,u}$
7.          use Weibull distribution to model $V_{i,u}$
8.          compute $\mu$ and $\sigma$ correspondingly
9.          compute $lb_{i,u}$ and $ub_{i,u}$
10.         add $lb_{i,u}$ and $ub_{i,u}$ to bin $b_i$
11.     $H$.append($h_u$)
12. return $H$
```

4.3 Threshold Calculation

After generating histogram models, HCAD adaptively determines the anomaly threshold for each category's histogram. The system first applies the generated model to training data and calculates anomaly scores for each bin as shown by Equation 3, where f is the frequency (i.e., event count) of that bin. lb is the lower bound and ub is the upper bound of the trained model.

$$score(v_i, b_j) = \begin{cases} \frac{|f_{i,j} - lb(v_i,b_j)|}{lb(v_i,b_j)}, & if\ f_{i,j} < lb(v_i, b_j) \\ \frac{|f_{i,j} - ub(v_i,b_j)|}{ub(v_i,b_j)}, & if\ f_{i,j} > ub(v_i, b_j) \\ 0, & otherwise \end{cases} \quad (3)$$

For a histogram h_i, the overall anomaly score is calculated as the maximum score of all its bins' scores.

$$score(h_i) = \max_{b_j \in h_i}(score(v_i, b_j)) \quad (4)$$

After computing the anomaly scores for all the historical data, HCAD extracts the largest score in histogram to compute the anomaly threshold for category v_i.

$$\delta(v_i) = \max_{h_i \in hist\_set(v_i)}(score(h_i)) * (1 + \varepsilon) \quad (5)$$

In Equation 5, ε is a parameter to represent the safe scale range. For instance, if $\varepsilon = 20\%$, it means the user believe that the score for real anomaly should be 120% larger than the historical maximum score in training data.

Algorithm 4: Online monitoring

Input: Stream data within sliding window w_{new}, histogram model H, histogram template T, anomaly threshold δ

Output: Anomaly label for w_{new}

```
1.  for each categorical value $v_u$ in $T$
2.      2D histogram $h_u$ = $T.get(v_u)$
3.      fill $h_u$ with the events in $w_{new}$
4.      $H_{new}$.append($h_u$)
5.  for each $h_u$ in $H_{new}$
6.      score($h_u$) = 0
7.      for each bin $b_i$ in $h_u$
8.          if unseen value or event duration
9.              comupute score($b_i$) by Eqs. 6, 7
10.         else
11.             compute score($b_i$) w.r.t. $H$
12.         if score($h_u$) < score($b_i$)
13.             score($h_u$) = score($b_i$)
14.     if score($h_u$) > $\delta(u)$
15.         return true
16. return false
```

4.4 Online Monitoring

Finally, HCAD uses the trained model to monitor streaming data from categorical sensors of deployed IoT. The main difficulty of online monitoring is on dealing with new values and unseen event durations: (1) Some categorical values may not appear in training data and there is no model for such values; (2) Some events may have very long or short durations that are outside the boundaries of histogram model.

To deal with these problems, HCAD has designed following strategies:

- If HCAD finds a new category value that not appeared in training data, it will directly assign a relatively high anomaly score.
- If HCAD finds an event with duration shorter than the minimum boundary of histogram model. The system will compute the anomaly score by Equation 6.

$$Score = \frac{|duration - lb(v_i,b_j)|}{lb(v_i,b_j)} \quad (6)$$

- If HCAD finds an event with duration larger than the maximum boundary. The system will compute the anomaly score by Equation 7.

$$Score = \frac{|duration - ub(v_i,b_j)|}{ub(v_i,b_j)} \quad (7)$$

Algorithm 4 describes the online monitoring process: First, HCAD fills the event data of new sliding window to histogram template and generates a new histogram for each categorical value (Lines 1 to 4). Then the algorithm matches the generated histograms to trained model and compute the anomaly score for each bin of the histogram (Lines 5 to 11). The overall score of the histogram is calculated as the maximum score of all the bins (Lines 12 to 13). If the score of any category's histogram is larger than threshold, the algorithm returns true, and an alert will be sent to the user (Lines 14 to 15). Otherwise, the algorithm returns false and continues monitoring the stream.

5 PERFORMANCE EVALUATION

In this section, a thorough experimental study for HCAD is conducted. First, we conduct the comparison experiment on HCAD and baselines. Then we evaluate some factors that affect the performance of proposed algorithm. We also present the ablation study of different components of HCAD. At last, we present a case study, followed by a short discussion on real applications of HCAD.

5.1 Experiment Setup

We evaluate the proposed HCAD method on three real datasets of different IoT applications, the statistic of datasets is listed in Table 2.

- Dataset D_1 [5]: The telemetry data from NASA Soil Moisture Active Passive (SMAP) satellite.
- Dataset D_2 [8]: The air quality data from an Italian city. The data were recorded from March 2004 to February 2005 by deployed chemical sensors. The raw data are numerical. There is no abnormal event in the data. We generated the categorial time series by mapping the numerical values into several keys and inserted 270 abnormal events for testing.
- Dataset D_3 [9]: The traffic data recorded from the twin cities metro area in Minnesota, collected by the deployed IoT sensors on streets around the city. The raw data are numerical. There is no abnormal event in the data. We generated the categorial time series by mapping the numerical values into several keys and inserted 78 abnormal events for testing.

We compare HCAD with the most recent state-of-the-art anomaly detection method for IoT sensors: LSTM with Non-parametric Dynamic Thresholding (LSTM-NDT) [5]. Table 3 lists the main parameters used in the experiments. For different datasets, LSTM-NDT requires the users to setup different anomaly threshold. However, HCAD does not require the parameter setup in advance (which is indeed hard for the user). It can use the same (default) parameter settings for all the datasets.

Table 2: Statistics of the datasets

Dataset	$\|R\|$	$\|R_{Train}\|$	$\|R_{Test}\|$	$\|E_N\|$	$\|E_A\|$	$\|\mathbf{R}_A\|$
D1	58670	23468	35202	1428	88	2350
D2	9358	5615	3743	355	270	530
D3	2501	1501	1000	134	78	103
$\|R\|$: total number of sensor reading; $\|R_{Train}\|$: training data size.						
$\|R_{Test}\|$: test data size; $\|E_N\|$: total number of normal events.						
$\|E_A\|$: total number of abnormal events; $\|R_A\|$: abnormal data size.						

Table 3: Experiment parameter setting

Dataset	Method	
	HCAD	LSTM-NDT
D_1	$m = 6$	$p = 0.13$
D_2	$\alpha_1 = 2.5, \beta_1 = 5$	$p = 0.03$
D_3	$\alpha_2 = 3, \beta_2 = 3$	$p = 0.03$
HCAD parapmeters: m: number of bins. α_1, β_1: determine left and right boundaries of histogram. α_2, β_2: determine lower and upper boundaries of normal range.		
LSTM-NDT parapmeters: p: anomaly threshold.		

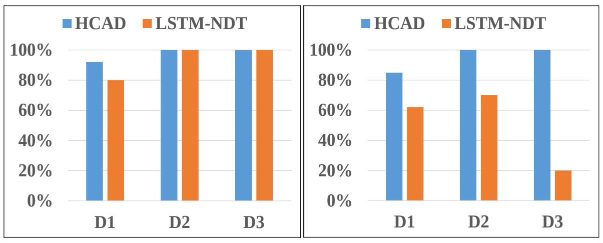
Table 4: Evaluation metrics

Measure	Formular
Precision	$\frac{\text{TP}}{\text{TP} + \text{FP}}$
Recall	$\frac{\text{TP}}{\text{TP} + \text{FN}}$
F1-Score	$2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$
TP: true positives, number of correctly detected anomalies. FP: false positives, number of false alters. FN: false negatives, number of missed anomalies.	

Table 4 shows the three metrics used for performance evaluation. Precision measures the robustness of anomaly detection method. Recall measures the sensitivity of anomaly detection method. F1-Score is a combined measure of precision and recall for an overall judgement.

5.2 Comparison on Accuracy and Efficiency

In the first experiment, we compare the proposed HACD with LSTM-NDT on the detection accuracy on different datasets. The precision and recall of these methods are shown in Figure 5. The precision of HCAD is 12% higher than LSTM-NDT on D_1. The recall of HCAD is 13% higher than LSTM-NDT on D_1, 30% higher on D_2, and 80% higher on D_3.



(A) Precision (B) Recall

Figure 5: Accuracy comparison on precision and recall

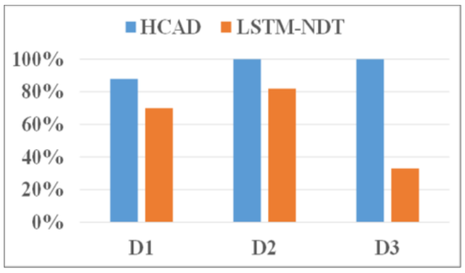


Figure 6: Accuracy comparison on F1 score

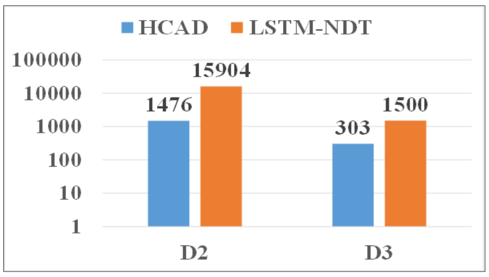


Figure 7: Comparison on detecting efficiency

As an overall measure, the F1 score of HCAD is 18% higher than LTSM-NDT on D_1 and D_2, and around 67% higher than LTSM-NDT on D_3, as shown in Figure 6.

In many IoT applications, the algorithm is required to detect the anomalies in real time. Hence, we compare the time cost of HCAD and LSTM-NDT in processing the test data stream and detecting anomalies. From the results in Figure 7, we can see that HCAD is much more efficient than LSTM-NDT: the time cost of HCAD is 1476 milliseconds on D_2 and 91% lower than LSTM-NDT; the time cost of HCAD is only 303 milliseconds on D_3 and 80% lower than LSTM-NDT.

5.3 Parameter Tuning and Ablation Study

In this experiment, we analyze the performance of HCAD by adjusting values of different parameters. We test the algorithm on the total dataset (i.e., Union of all three datasets). The parameters of HCAD are α_1, β_1,

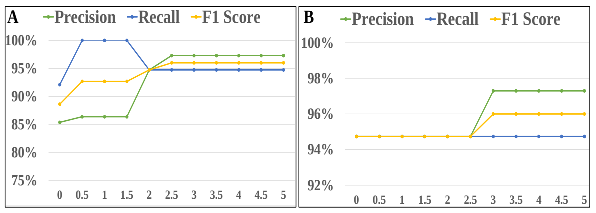


Figure 8: Accuracy. (A) shows influence of α_1. (B) shows influence of β_1

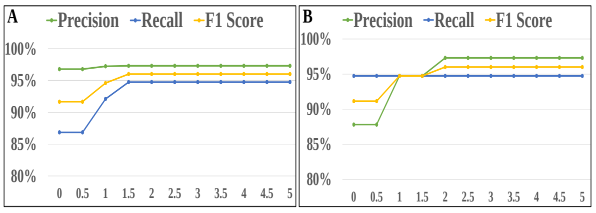


Figure 9: Effectiveness. (A) shows influence of α_2. (B) shows influence of β_2

α_2, and β_2, as listed in Table 3. We adjust them in training stage to generate different models. Then we apply the trained models to testing data for anomaly detection. We use precision, Recall, and F1-score to evaluate the influence of parameters.

First, we study influence of α_1. α_1 is to determine the minimum bound of histogram model. we increase α_1 from 0 to 5 with a stride equal to 0.5. F1-score of HCAD is converged when α_1 equals to 2.5 (Figure 8 (A)). So we set α_1 as 2.5 as the default value. β_1 is used to determe the maximum bound of histogram model. We use same procedure to increase β_1 from 0 to 5. The influence of β_1 on precision is greater than recall (Figure 8(B)). The F1-score is converged when β_1 equals to 3.

Then we study influence of α_2. α_2 is used to control the lower bound of frequency. Figure 9 (A) shows that the influence of α_2 on recall is greater than precision. The F1-score converges at the value around 1.5. Finally, we study the influence of β_2, which controls the upper bound of frequency. Figure 9 (B) shows that the influence of β_2 on precision is greater than recall. The F1-score converges when β_2 equals to 2.

Table 5 shows the ablation study on the impact of adaptive sliding window (ASW), 3D histogram (3DH), and Weibull distribution (WD). We use dataset D_1 to analyze the performance of HCAD with the different combinations of components.

In the first round, we remove all the components including ASW, HECV, and WD: (1) We use a fixed sliding window (The length of sliding with is set as 100) to replace the adaptive sliding window; (2) we use a 2D histogram to replace the 3D histogram, and (3) we use the normal distribution to replace Weibull distribution. We record the performance of such a model as baseline.

Table 5: Ablation study of components

Components	Different combinations of components			
ASW	×	√	√	√
3DH	×	×	√	√
WD	×	×	×	√
Precision (%)	47%	50%	82%	92%
Recall (%)	54%	54%	75%	85%
F1 Score (%)	50%	52%	78%	88%

In the second round, we add ASW to HCAD and compare it with the baseline. The performance is slightly improved (~2% increase in F1-score). In the third round, ASW and HECV are both added to HCAD. The accuracy is significantly boosted (~26% increase in F1-score). Finally, all components are added, and the performance is further improved. The table shows that the 3D histogram model (3DH) is the key component to improve detection accuracy of HCAD.

5.4 Case Study and Discussions

In this section, we use a case study to illustrate the reasons that why HCAD has better performance than LSTM-NDT.

Figure 10 (A) shows a categorical sensor dataset. The data are divided to two parts by a green line. The left part is used for training and the right part is used for testing. For simplicity, we select a dataset with only two unique values (0 and 1). The testing data contain one abnormal event. Figure 10 (B) shows details of this abnormal event. The anomaly is caused by frequent switching of category “0” and “1” in a short time period.

LSTM-NDT uses historical data to predict new value. If the different between prediction and observations is larger than a certain threshold, the system will report anomaly. Unfortunately, it is difficult for LSTM-NDT to correctly predict such categorical values.

On the other hand, HCAD not only considers the value of categories but also considers the switching frequency of them. It transfers the streaming data from original 2D space to a 3D histogram. Figure 11 shows the histogram of category “1”. There are four bins in the histogram. The frequencies of four bins are [21,2,0,0] and the corresponding thresholds are {[0,8], [0,4], [0,2], [0,2]}. The frequency of bin-1 is outside of the corresponding threshold, which is therefore considered as an abnormal data by HCAD.

Figure 12 shows a snapshot of the prototype system that we developed to monitor categorical sensor data for IoT. The green lines in Figure 12 (A) shows all the received data from a specific sensor. The lines in Figure 12 (B) are the new data in current sliding window. The 3D objects in Figure 12 (D) are the histogram constructed from the new streaming data. The lines in Figure 12 (C) are the calculated anomaly scores. The red lines indicate the detected abnormal events.

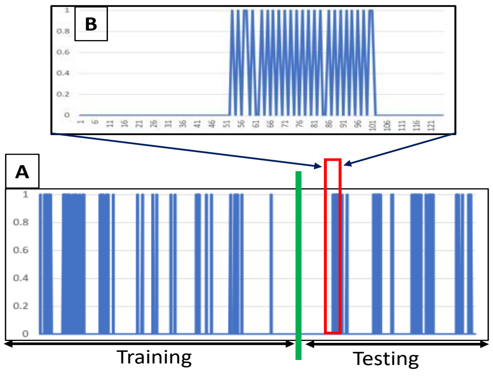


Figure 10: (A) shows a categorical sensor dataset. The data to the left of green line is used for training and the rest is for testing. The red rectangle is sliding window. (B) shows details in sliding window.

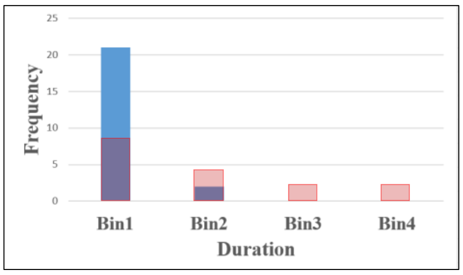


Figure 11: Blue rectangles are frequencies of category “1”. Red rectangles represent thresholds of bins.

To apply HCAD in real IoT systems, the solution can be deployed in a distributed manner. The model training is conducted in the cloud. Once the histogram model is generated, it is sent to every edge device of the IoT. The online monitoring algorithm (Algorithm 4) is running on the edge device. Since Algorithm 4 only needs to store the current window of streaming data and train histogram models with limited memory. Once an anomaly is detected, the edge will communicate with the cloud and report the abnormal events. In this way, the edge does not need to communicate with the cloud during normal period, the cost of bandwidth is limited. HCAD tool is programmed with C/C++ language. There are no special requirements for hardware, and it can be easily embedded into IoT operating system such as Raspberry Pi.

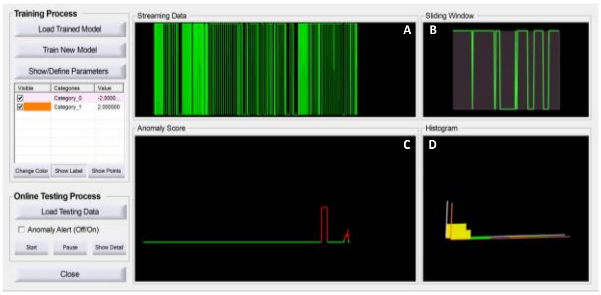


Figure 12: The GUI interface of HCAD based IoT monitoring system. (A) stores all the received data. (B) plots the current window. (C) shows the anomaly score. (D) is 3D histogram.

6 Conclusion and Future Work

In this study, we present a 3D Histogram based Categorical Anomaly Detection (HCAD) method for monitoring the sensor data in IoT devices. In the proposed method, the system generates a 3D histogram model on the dimensions of category, event duration, and frequency. HCAD automatically determines normal ranges of the histogram model and anomaly thresholds. Then, the system applies the model to monitor streaming data and detect anomalies in real time. Extensive experiments on real datasets show that HCAD achieves higher accuracy and efficiency in anomaly detection tasks than other baselines.

In the near future, we plan to extend HCAD to complex IoT monitoring with both categorical and numerical sensors, and to apply HCAD in other application domains such as weather forecasting and financial analysis.

Acknowledgements

The authors would like to thank Matthew Horak and Sadananda Narayanappa from Lockheed Martin Space for constructive comments and suggestions.

References

[1] A. Shabtai, “Anomaly detection using the knowledge-based temporal abstraction method,” *arXiv preprint arXiv:1612.04804*, 2016

[2] Y. Shahar, “A framework for knowledge-based temporal abstraction,” *Artificial Intelligence*, vol. 90, no. 1–2, pp. 79-133, 1997

[3] T. Yairi, M. Nakatsugawa, K. Hori, S. Nakasuka, and K. Machida, “Adaptive limit checking for spacecraft telemetry data using regression tree learning,” *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No.04CH37583)*, vol.6, pp. 5130-5135, 2004

[4] Y. Gao, T. Yang, M. Xu, N. Xing, “An unsupervised anomaly detection approach for spacecraft based on normal behavior clustering,” *2012 Fifth International Conference on Intelligent Computation Technology and Automation*, pp. 478-481, 2012

[5] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Soderstrom, “Detecting spacecraft anomalies using LSTMs and nonparametric dynamic thresholding,” *In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 387-395, 2018.

[6] M. Schwabacher, N. Oza, B. Matthews, “Unsupervised anomaly detection for liquid-fueled rocket propulsion health monitoring,” *Journal of aerospace computing, information, and communication*, vol. 6, no. 7, pp. 464–482, 2009

[7] L. Tang, B. Cui, H. Li, G. Miao, D. Yang, X. Zhou, “Effective variation management for pseudo periodical streams,” *In Proceedings of the 2007 ACM SIGMOD international Conference on Management of Data*, pp. 257-268, 2007

[8] UCI Machine Learning Repository, “Air Quality Data Set from https://archive.ics.uci.edu/ ml/datasets/air+quality,” *UCI Machine Learning Repository*

[9] Numenta Anomaly Benchmark (NAB), “Real time traffic data from https://github.com/numenta/ NAB/tree/master/data,” *Numenta Anomaly Benchmark (NAB)*

[10] A. Arora, P. Dutta, S. Bapat, “A line in the sand: awireless sensor network for target detection, classification, and tracking,” *Computer Networks*, vol. 46, no. 5, pp. 605-634, 2004.

[11] R. Szewczyk, J. Polastre, J. Mainwaring, “Lessons from a sensor network expedition,” *In European Workshop on Wireless Sensor Networks*, Springer, pp. 307-322, 2004.

[12] P. Buonadonna, D. Gay, JM. Hellerstein, W. Hong, S. Madden, “TASK: Sensor network in a box,” *In Proceeedings of the Second European Workshop on Wireless Sensor Networks,2005*, IEEE, pp. 133-144, 2005

[13] H. Rinne, “The Weibull distribution: a handbook,” *Chapman and Hall/CRC*, 2008

[14] J. Li, S. Di, Y. Shen, and L. Chen, “FluxEV: a fast and effective unsupervised framework for time-series anomaly detection,” *In Proceedings of*

14th ACM International Conference on Web Search and Data Mining, pp. 824-832, 2021

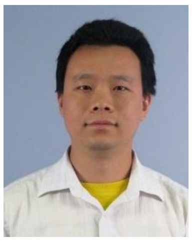
[15] C. Aytekin, X. Ni, F. Cricri, and E. Aksu, "Clustering and unsupervised anomaly detection with l_2 normalized deep auto-encoder representations," *In 2018 International Joint Conference on Neural Networks (IJCNN)*, IEEE, pp. 1-6, 2018

[16] J. Gao, X. Song, Q. Wen, P. Wang, L. Sun, H. Xu, "Robusttad: Robust time series anomaly detection via decomposition and convolutional neural networks," *arXiv preprint arXiv:2002.09545*, 2020

[17] T. Wen, R. Keyes, "Time series anomaly detection using convolutional neural networks and transfer learning," *arXiv preprint arXiv: 1905.13628*, 2019

AUTHOR BIOGRAPHIES

Peng Yuan is an Associate Researcher in Data Science and System Research Dept. at NEC Laboratories America. His main research interests include anomaly and outlier detection in IoT and CPS.



Dr. Lu-An Tang is a Senior Researcher in Data Science and System Research Dept. at NEC Laboratories America. His main research interests include anomaly and outlier detection in IoT and CPS, spatial-temporal data mining, graph data mining and AI based cyber security. His research results have appeared in premier data management and data mining venues (e.g., VLDB, SIGMOD, ICDE, KDD, ICDM, SDM). He holds a PhD degree in Computer Science from University of Illinois at Urbana-Champaign.

Dr. Haifeng Chen is currently the head of Data Science and System Security Department at NEC Laboratory America, Princeton, New Jersey. He received the BEng and MEng degrees in automation from Southeast University China, and the PhD degree in computer engineering from Rutgers University, New Jersey, in 2004. His research focus is on big data analytics, AI, software and system security, smart service and platforms. He has co-authored over a hundred publications in top conference and journals, and has over 70 patents granted.

Moto Sato Moto Sato is a director of business development at NEC Laboratories America. He was the director of PC silicon valley center and venture liaison of NEC Systems. He has coordinated and directed multiple research projects of IoT between NEC and its partners, including the failure detection for autonomous driving, sensing with optical devices, anomaly detection on satellite sensors, etc.

Kevin Woodward is a Fellow Emeritus with Lockheed Martin Space, and manages the company's Business Innovation, Transformation and Enterprise Excellence artificial intelligence/machine learning portfolio. For nearly four decades, he has worked within the aerospace and defense industry in the areas of artificial intelligence, predictive analytics, model-based engineering, affordability management, and engineering economics. He has supported and led U.S. domestic and foreign military, civil, and commercial projects in the realms of advanced manned and unmanned aircraft, missiles, launch vehicles, spacecraft, software, and information technology. A recipient of Lockheed Martin's highest and most prestigious honor - the NOVA Award - for innovations pertaining to the F-35 advanced fighter aircraft concept development, Kevin organized and currently leads an international artificial intelligence collaboration, holds an advanced degree in business administration (MBA) from the University of Southern California, as well as bachelor's degrees in International Relations (National Security Affairs) and in Japanese from Brigham Young University.